

Reverse Retrieval Augmented Generation

Client-Side Context Injection for Small Language Models

How live DOM extraction makes 8B models punch above their weight class.

russell@unturf, cthegray, TimeHexOn, foxhop

uncloseai.com · permacomputer.com

February 2026

Abstract

Traditional Retrieval Augmented Generation (RAG) requires a server-side pipeline: chunk documents, embed them into vectors, store them in a database, & retrieve by similarity search at query time. This architecture demands infrastructure, indexing latency, & maintenance of embedding models & vector stores.

Reverse Retrieval Augmented Generation (Reverse RAG) inverts this entirely. Instead of the server fetching documents to augment the prompt, the client extracts live content from the page the user currently views & injects it directly into the conversation context. The data comes to the model. No vector database. No embeddings. No indexing pipeline. No server-side retrieval.

uncloseai.js implements this technique as an AGPL-3.0-only algorithm in a single-file JavaScript library that adds a machine learning chat interface to any webpage. By feeding the model the full, fresh content of whatever page the user visits, small 8B-parameter models produce answers that rival much larger models on page-specific questions.

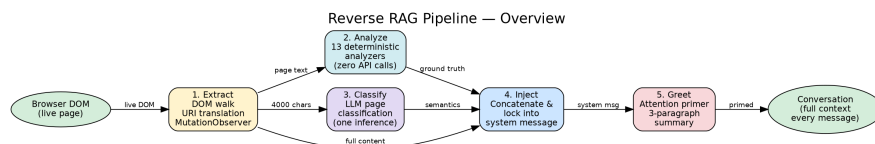


Figure 1: The five-stage Reverse RAG pipeline. Content flows from the browser DOM through extraction, analysis, classification, & injection into a locked system message.

1. The Problem with Traditional RAG

Standard RAG systems follow a retrieval-then-generate pattern:

1. **Ingest:** Crawl documents, split into chunks of ~512 tokens
2. **Embed:** Run each chunk through an embedding model (e.g., OpenAI text-embedding-3, sentence-transformers)
3. **Store:** Insert vectors into a database (Pinecone, Weaviate, ChromaDB, pgvector)
4. **Query:** When a user asks a question, embed the query, find top-k similar chunks by cosine similarity
5. **Generate:** Stuff the retrieved chunks into the prompt, send to the LLM

This pipeline carries real costs:

- **Stale data:** Documents require re-crawling, re-chunking, & re-embedding to stay current. Most RAG pipelines lag hours or days behind.
- **Retrieval failures:** Cosine similarity does not equal understanding. The "most similar" chunk often misses the most relevant one. Critical context gets dropped.

- **Infrastructure overhead:** Vector databases, embedding endpoints, chunking pipelines, reindexing jobs. Each adds a point of failure & a monthly bill.
- **Context fragmentation:** Chunking destroys document structure. A paragraph retrieved without its heading, or a code block without its explanation, loses meaning.
- **Cold start:** New documents remain unavailable until the ingestion pipeline processes them. For rapidly changing content, this lag proves unacceptable.

2. Reverse RAG: The Client Has the Context

Reverse RAG starts from a different observation: **the user already looks at the document they want to ask about.** The browser holds the full, rendered, up-to-the-second content right there in the DOM. Why retrieve it again from a database?

The Algorithm

1. **Extract:** Walk the live DOM tree. Pull text, links (as markdown), metadata, structured data. Wait for dynamic content to settle (MutationObserver with 500ms quiet period).
2. **Analyze:** Run 13 deterministic analyzers on the extracted content. Zero API calls. Compute reading metrics, readability scores, code block detection, link topology, entity patterns, media inventory, form detection, & more.
3. **Classify:** Send a 4000-character preview to the model for one-shot page classification: type, author, topics, tone, domain, audience, key phrases, freshness.
4. **Inject:** Concatenate the computed intelligence, classification, & full page content into the system message. Lock it in for the entire conversation session.
5. **Converse:** Every subsequent user message carries the full page context already in the system prompt. The model maintains complete knowledge of the page at all times.

No step involves a server fetching documents. No vector similarity search runs. The context always comprises the *entire* page, not a "most relevant" fragment chosen by an embedding model that might get it wrong.

3. Architecture

Stage 1: Content Extraction

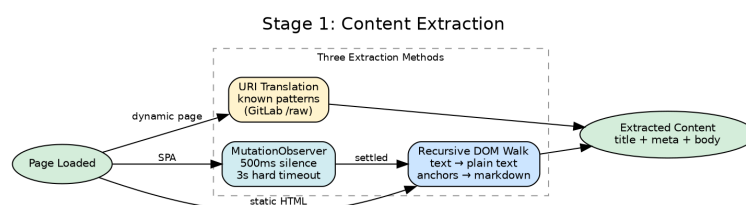


Figure 2: Content extraction flow. URI translation handles known dynamic patterns; MutationObserver handles SPAs; recursive DOM walk produces clean text with markdown links.

The extraction pipeline handles static HTML, dynamic SPAs, & server-rendered content through three mechanisms:

URI Translation: Known dynamic page patterns (GitLab CI logs, API documentation portals) map to their raw content endpoints. A GitLab job page fetches /raw instead of parsing the rendered HTML. This handles cases where the visible DOM serves as a thin shell over data loaded asynchronously.

DOM Settlement: A MutationObserver watches for DOM changes. Extraction waits until 500ms of silence (no mutations), with a hard timeout at 3 seconds. This handles React/Vue/Svelte apps that hydrate after initial page load.

Recursive DOM Walk: The traversal covers the full document body. Text nodes become plain text. Anchor tags become markdown links: [link text](href). The output preserves page structure without HTML noise.

```
// Simplified extraction logic
function extractDOMContent() {
  const title = document.title;
  const meta = document.querySelector('meta[name="description"]')?.content;
  const body = walkDOM(document.body); // recursive text + markdown links
  return `**Page Title**: ${title}\n\n${meta}\n\n${body}`;
}
```

Stage 2: Page Intelligence (13 Deterministic Analyzers)

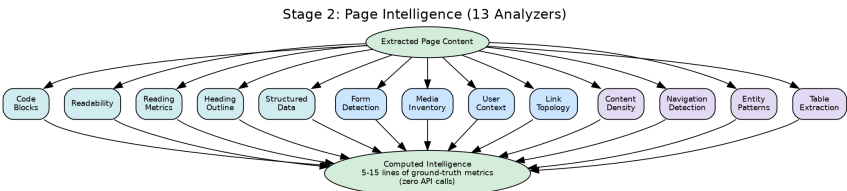


Figure 3: All 13 analyzers run in parallel on the extracted content. Each produces ground-truth metrics with zero API calls. Failed analyzers get omitted; others continue.

Before any model call, 13 analyzers extract ground-truth metrics from the page. Every analyzer runs locally in the browser. Zero network requests. Zero tokens consumed.

Analyzer	Output	Purpose
Structured Data	Open Graph, JSON-LD, Twitter Cards, canonical URI, author, publish date	Machine-readable page metadata
Heading Outline	h1-h6 hierarchy with text	Document structure map
Reading Metrics	Word count, sentence count, paragraph count, reading time (238 wpm Brysbaert 2019)	Content scope estimation
Readability	Flesch-Kincaid grade level with descriptor	Audience calibration
Code Blocks	Count, languages detected, total lines, code-to-prose ratio	Technical content identification
Link Topology	Internal vs. external count, top 5 external domains	Reference network understanding
User Context	Timezone, browser language, device type, referrer	Personalization signals
Media Inventory	Image count, alt-text coverage, video embeds (YouTube/Vimeo)	Multimedia awareness
Form Detection	Form count, classified type (login/search/contact/checkout)	Interactive element awareness

Table Extraction	Up to 5 tables with headers & row counts	Structured data in prose
Entity Patterns	Emails, prices, dates, version numbers, IPs, percentages (regex, max 10 each)	Factual anchor points

The formatter optimizes output for token efficiency. Empty sections get omitted. A typical page produces 5-15 lines of computed intelligence:

```
Reading: 2,341 words | 89 sentences | 34 paragraphs | ~10 min read
Readability: Flesch-Kincaid grade 9.2 (high school)
Code: 3 blocks (python, javascript) | 156 lines | 23% code-to-prose
Links: 42 internal, 8 external | top: github.com(3), stackoverflow.com(2)
Structured Data: og:type=article | author=fxhp | published=2025-01-15
Entities: prices: $99.99, $129.99 | versions: v1.2.3, v2.0.0
```

Stage 3: LLM Page Classification

A single inference call classifies the page. The model receives the first 4000 characters of extracted content & returns a JSON object:

```
{
  "type": "documentation",
  "author": "fxhp",
  "publishDate": "2025-02-24",
  "topics": ["machine learning", "inference"],
  "entities": ["vLLM", "Hermes", "Qwen"],
  "tone": "technical",
  "domain": "technology",
  "audience": "developers running local inference",
  "keyPhrases": ["dynamic model discovery", "streaming SSE"],
  "summary": "Documentation for running local LLM inference with vLLM",
  "contentLanguage": "en",
  "freshness": "evergreen"
}
```

This classification remains optional. On failure, the conversation proceeds with computed intelligence & raw content alone. The system never blocks on a failed classification.

Stage 4: Context Injection & Locking

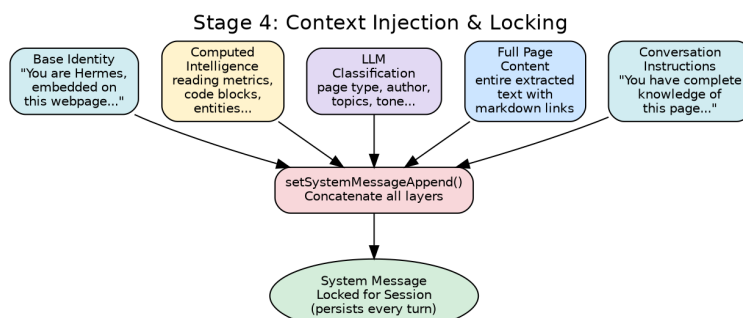


Figure 4: Five layers concatenate into the system message via `setSystemMessageAppend()`. The combined context locks for the entire session.

All three layers concatenate into the system message:

```
SYSTEM MESSAGE:
[Base identity: "You represent Hermes, embedded on this webpage..."]
[Computed intelligence: reading metrics, code blocks, entities...]
[Classification: page type, author, topics, tone, key phrases...]
```

```
[Full page content: entire extracted text with markdown links]
[Conversation instructions: "You possess complete knowledge of this page..."]
```

This combined context gets set once via `setSystemMessageAppend()` & persists for the entire conversation session. Every subsequent user message carries the full page context in the system prompt. The model never loses sight of what page it occupies.

Stage 5: Greeting as Attention Primer

The model generates a 3-paragraph greeting that demonstrates page understanding: what the page covers, what stands out most, & how it can help. This greeting serves as an attention primer. By forcing the model to summarize the page before the user asks anything, the model's internal representations already align with the page content when the first real question arrives.

4. Why Small Models Punch Above Their Weight

An 8B-parameter model with the right context in its system prompt outperforms a 70B model that guesses. This represents the core insight of Reverse RAG.

Traditional RAG gives the model fragments: 3-5 chunks of ~512 tokens each, selected by embedding similarity, ripped from their surrounding context. The model must reconstruct meaning from these fragments while also answering the user's question.

Reverse RAG gives the model *everything*: the full page text, the heading structure, the link network, code block languages, reading level, entity patterns, & a classification of what kind of page it reads. The model needs to infer nothing about the page. It all sits right there.

For page-specific questions ("what does this function do?", "who wrote this?", "summarize the third section"), context completeness beats parameter count. A small model with perfect context outperforms a large model with partial context.

The tradeoff stands clear: Reverse RAG consumes more prompt tokens per message (the full page sits in every system prompt). But inference on small models runs cheap. The tokens spent on context deliver far more value than the infrastructure costs of running a RAG pipeline that produces worse context.

5. Two-Layer Context: Ground Truth + Semantic Understanding

The 13 deterministic analyzers & the LLM classification serve different roles:

Layer 1: Computed Intelligence (ground truth). The model cannot hallucinate these facts because they come computed directly from the DOM. The page contains exactly 2,341 words. The Flesch-Kincaid grade reaches exactly 9.2. Exactly 3 code blocks exist in Python & JavaScript. The model receives these as pre-computed facts & can cite them with confidence.

Layer 2: LLM Classification (semantic understanding). The model's own classification of the page type, tone, audience, & key phrases. This layer stays subjective & can miss, but it primes the model's attention toward the right framing. A page classified as "recipe" triggers different conversational patterns than one classified as "documentation."

Together, these layers give the model both *what the page contains* (computed) & *what the page means* (classified). Neither layer alone suffices. Ground truth without semantic framing produces dry, unfocused answers. Semantic framing without ground truth produces confident but potentially wrong answers.

6. Reverse RAG vs. Traditional RAG

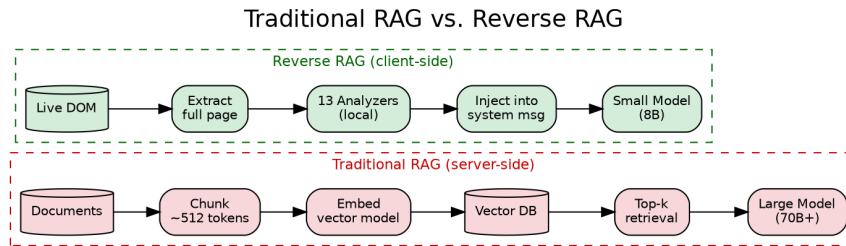


Figure 5: Architecture comparison. Traditional RAG requires five server-side components; Reverse RAG runs entirely in the browser with four client-side stages.

Dimension	Traditional RAG	Reverse RAG
Data flow	Server retrieves documents for the model	Client pushes page content to the model
Infrastructure	Vector DB, embedding model, chunking pipeline, reindexing jobs	None. Runs in the browser.
Freshness	Hours to days behind (reindex lag)	Real-time. Extracted from live DOM.
Context scope	Top-k chunks (~2500 tokens)	Entire page + computed intelligence
Context quality	Fragments selected by cosine similarity (can miss critical content)	Complete page with structure preserved
Retrieval failures	Common. Embedding similarity does not equal understanding.	Impossible. The pipeline includes the entire page.
Token cost per query	Lower (only retrieved chunks)	Higher (full page in system prompt)
Best model size	Large (must reason over fragments)	Small (8B suffices with full context)
Use case	Question answering over large document corpora	Contextual assistance on the page you view
Setup time	Days to weeks (pipeline, embeddings, tuning)	One script tag. Done.

These techniques do not compete. Traditional RAG excels at searching across thousands of documents. Reverse RAG excels at deep understanding of the one document the user actively reads. They solve different problems.

7. Implementation

[uncloseai.js](#) implements Reverse RAG as an AGPL-3.0-only algorithm. The complete source remains available & auditable. The implementation spans these files:

File	Role
<code>content.js</code>	DOM extraction, URI translation, page analysis prompts
<code>page-intelligence.js</code>	13 deterministic analyzers (zero API calls)
<code>config.js</code>	System message assembly, context concatenation
<code>chat.js</code>	Message delivery with token budget management
<code>uncloseai-embed-modal.js</code>	Pipeline orchestration, greeting generation

Installation

```
<script src="https://uncloseai.com/uncloseai.js" type="module"></script>
```

One line. The Reverse RAG pipeline runs automatically when the user opens the chat modal. No configuration required. The model receives the full page context on the first interaction.

8. Graceful Degradation

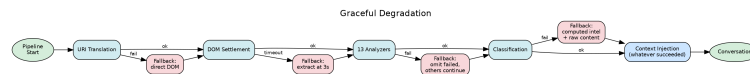


Figure 6: Every stage fails independently. The pipeline always produces a usable context, even when individual stages encounter errors.

Every stage of the pipeline fails independently:

- If URI translation fails: falls back to DOM extraction
- If DOM still loads: extracts what exists after 3 seconds
- If any of the 13 analyzers throws: that analyzer's output gets omitted, others continue
- If page classification fails: conversation proceeds with computed intelligence + raw content
- If the page lacks meaningful content: model acknowledges this in the greeting

The system never blocks on a failure. A partial context always beats no context.

9. Token Budget Management

Full page injection consumes prompt tokens. The system manages this with adaptive token budgeting:

- **Large context models (>32k tokens):** Reserve 10% or 2k tokens for the response, whichever proves smaller. The full page fits easily.
- **Medium context (8k-32k):** Reserve 20% or 1.5k tokens. Most pages fit. Very long pages may have their content naturally bounded by the 4000-char classification preview.
- **Small context (<8k):** Reserve 50% of estimated input tokens. Page content gets included as-given; the model works with what fits.

This adaptive strategy ensures the model always retains room to generate a meaningful response, even when the page content grows large relative to the context window.

10. License

The entire unclosetai.js library carries the **AGPL-3.0-only** license: the Reverse RAG pipeline (content extraction, page intelligence, context injection), the chat interface, TTS, translation, & vault encryption. You may use, study, & modify the code, but any networked deployment of modified versions must release the source under the same license. This keeps the technique open & auditable.

Citation

russell@unturf, cthegray, TimeHexOn, foxhop. "Reverse Retrieval Augmented Generation: Client-Side Context Injection for Small Language Models." unclosetai.com, 2026. <https://unclosetai.com/reverse-rag.html>

License



GNU Affero General Public License v3

AGPL-3.0-only

PERMACOMPUTER PREAMBLE - NO WARRANTY

This is free software for the public good of a permacomputer hosted at permacomputer.com - an always-on computer by the people, for the people. One which is durable, easy to repair, and distributed like tap water for machine learning intelligence.

The permacomputer is community-owned infrastructure optimized around four values:

TRUTH	- Source code must be open source & freely distributed
FREEDOM	- Voluntary participation without corporate control
HARMONY	- Systems operating with minimal waste that self-renew
LOVE	- Individual rights protected while fostering cooperation

This software contributes to that vision by making machine learning accessible to everyone through a free, open, embeddable chat interface. Code is seeds to sprout on any abandoned technology.

Learn more: <https://www.permacomputer.com>

NO WARRANTY. THE SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND.

That said, our permacomputer's digital membrane stratum continuously runs unit, integration, and functional tests on all of it's own software - with our permacomputer monitoring itself, repairing itself, with minimal human in the loop guidance. Our agents do their best.

Copyright (C) 2025-2026 TimeHexOn & foxhop & cthegray & russell@unturf
<https://www.timehexon.com>
<https://www.foxhop.net>
<https://carltonthegray.com>
<https://www.unturf.com/software>
<https://www.permacomputer.com>
<https://unclosetai.com>
<https://russell.ballestrini.net>

GNU AFFERO GENERAL PUBLIC LICENSE
Version 3, 19 November 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<https://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies

of this license document, but changing it is not allowed.

Preamble

The GNU Affero General Public License is a free, copyleft license for software and other kinds of works, specifically designed to ensure cooperation with the community in the case of network server software.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, our General Public Licenses are intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

Developers that use our General Public Licenses protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License which gives you legal permission to copy, distribute and/or modify the software.

A secondary benefit of defending all users' freedom is that improvements made in alternate versions of the program, if they receive widespread use, become available for other developers to incorporate. Many developers of free software are heartened and encouraged by the resulting cooperation. However, in the case of software used on network servers, this result may fail to come about. The GNU General Public License permits making a modified version and letting the public access it on a server without ever releasing its source code to the public.

The GNU Affero General Public License is designed specifically to ensure that, in such cases, the modified source code becomes available to the community. It requires the operator of a network server to provide the source code of the modified version running there to the users of that server. Therefore, public use of a modified version, on a publicly accessible server, gives the public access to the source code of the modified version.

An older license, called the Affero General Public License and published by Affero, was designed to accomplish similar goals. This is a different license, not a version of the Affero GPL, but Affero has released a new version of the Affero GPL which permits relicensing under this license.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU Affero General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a

covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display

Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular

product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or

requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a

covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a

covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Remote Network Interaction; Use with the GNU General Public License.

Notwithstanding any other provision of this License, if you modify the Program, your modified version must prominently offer all users interacting with it remotely through a computer network (if your version supports such interaction) an opportunity to receive the Corresponding Source of your version by providing access to the Corresponding Source from a network server at no charge, through some standard or customary means of facilitating copying of software. This Corresponding Source shall include the Corresponding Source for any work covered by version 3 of the GNU General Public License that is incorporated pursuant to the following paragraph.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the work with which it is combined will remain governed by version 3 of the GNU General Public License.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU Affero General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU Affero General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the

GNU Affero General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU Affero General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

This program is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Affero General Public License for more details.

You should have received a copy of the GNU Affero General Public License along with this program. If not, see <<https://www.gnu.org/licenses/>>.

Also add information on how to contact you by electronic and paper mail.

If your software can interact with users remotely through a computer network, you should also make sure that it provides a way for users to get its source. For example, if your program is a web application, its interface could display a "Source" link that leads users to an archive of the code. There are many ways you could offer source, and different solutions will be better for different programs; see section 13 for the specific requirements.

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU AGPL, see <<https://www.gnu.org/licenses/>>.