Lemmy Project Structure

A link aggregator and forum for the decentralized web. A high-performance Rust application built with Actix-web and Diesel, focusing on ActivityPub federation.

Updated 2025-12-30

#lemmy #rust #actix-web #diesel #activitypub #federation #reddit-alternative

PNG PDF Copy Prompt

Project Directory

```
lemmy/
  crates/ Cargo Workspace
    server/ Main Applicatio…
    api/ REST API Handle…
    apub/ ActivityPub Pro…
    db_schema/ Diesel Schema &…
    db_views/ Postgres View d…
  migrations/ SQL Migrations …
  docker/ Deployment & Fe…
  Cargo.toml
  diesel.toml
```

Repository Info

Repository - LemmyNet/lemmy

Stars - 10k+

License - AGPL-3.0

Last Analyzed - December 2025

Tech Stack

Language - Rust

Web Framework - Actix-web

ORM - Diesel

Database - PostgreSQL

Protocol - ActivityPub

Format - JSON-LD

Architecture Notes

Lemmy is built as a modular **Cargo Workspace**. It relies heavily on **Diesel** for a type-safe interface to PostgreSQL. A distinct feature of Lemmy's architecture is its heavy use of **PostgreSQL Views** (`db_views`) to handle complex aggregations (like calculating hot-ranking scores or unread counts) efficiently at the database level. Federation is handled by the `apub` crate, which implements the ActivityPub standard.

Key Directories

`crates/apub/` - The federation engine. Handles sending and receiving activities (Like, Post, Follow) between Lemmy instances.

`crates/db_views/` - Contains definitions for materialized and non-materialized views that simplify complex queries for the API.

`crates/api/` - The Actix-web server implementation. It defines the JSON-RPC-like REST API that the frontend (`lemmy-ui`) consumes.

Why This Structure?

Lemmy is a prime example of a 'Social Systems' architecture in Rust. It shows how to combine high-performance web frameworks with a robust, type-safe database layer to build a decentralized platform.