Project Overview: AION Classic EU Companion Platform

This project is a comprehensive, integrated companion platform for AION Classic EU, designed to address the scarcity of structured resources for this niche MMO. The platform will centralize and deeply structure game data, community knowledge, and player tools into a unified ecosystem. Its core objective is to enhance player experience through organized information, planning utilities, and an intelligent assistant, ultimately fostering a more informed and efficient player community.

System Architecture & Core Components

1. Data Repository (Structured Game Knowledge Base)

The foundational layer of the platform. It moves beyond simple wikis by implementing a highly normalized relational database to model the game's entities with deep, query-friendly structure.

- **Example - Quest System:** A quest record will not only contain basic metadata (title, description, NPC, type) but will be linked to detailed reward entities.
- **QuestRewards Table:** Captures all reward types (XP, Kinah, AP, items).
- **Junction Tables:** Manage many-to-many relationships (e.g., `QuestRewards_Items`), allowing precise querying such as "all quests awarding a specific item" or "quests available to a Level 58 Elyos Gladiator."
- **Scope:** This structured approach will extend to all key game entities: Items, Mobs, Skills, Locations, Crafting Recipes, etc.

2. Structured Guides System

A dedicated module for tactical and strategic content, enforcing a consistent schema to ensure clarity, comparability, and machine-readability.

- **Format:** Guides will be composed of structured fields (e.g., Objective, Required Level/Class, Recommended Gear, Step-by-Step Phases, Loot Tables) rather than free-form blog posts.
- **Integration:** Will be directly linked to relevant entities in the Data Repository (e.g., linking a dungeon guide to the specific items dropped within it).

3. Integrated Blog Platform

A traditional blog for community news, updates, and editorial content, enhanced with deep platform integration.

- **Feature:** Authors can seamlessly reference and embed entities from the Data Repository or Guide system (e.g., "This strategy uses the [Sword of Example] and complements our [Abyss Entry Guide]").
- **Purpose:** Serves as the dynamic, human-facing counterpart to the structured database.

4. Player Routines & Activity Manager

A unique productivity module to help players plan and track in-game goals.

- **Activities:** User-defined units representing an "Action + Target" (e.g., "Complete [Quest: X]", "Participate in [Fortress Siege]", "Craft [Item: Y]").
- **Routines:** Compositions of Activities scheduled as `RoutineItems` with additional planning data (e.g., `CalendarEntry`, estimated time, weekly recurrence).
- **Utility:** Enables players to build personalized weekly checklists, track progress, and optimize their playtime.

5. Multi-Channel Notification System

A service layer to keep users engaged and informed.

- **Channels:** Supports browser notifications, email, and Discord webhooks.
- **Triggers:** Notifications for new relevant blog posts, routine reminders, or system-wide announcements.

6. AI Assistant Service

The intelligent interface to the entire platform, leveraging the structured data to provide precise, actionable answers.

Capabilities

- **Query Engine:** Uses custom tools to interrogate the Data Repository and Guides (e.g., "Find high-XP quests for a Level 58 Elyos").
- **Routine Generation:** Parses natural language user goals ("I want to catch up to my level 62 friends") and uses the Routines API to create a tailored set of weekly activities with time estimates.
- **Contextual Help:** Integrates recent blog posts and guide updates into its responses for current advice.

Example Interaction: A user prompt like "I am an Elyos gladiator at level 58... what quests should I do and how much time per week?" would trigger a query for level-appropriate quests, an analysis of their reward efficiency, and the automated creation of a suggested weekly routine with a time commitment summary.

7. Community-Driven Content Management & Governance

To ensure the platform's comprehensiveness and sustainability, core systems are designed as community-driven repositories with robust collaboration and quality control mechanisms. This model transforms the platform into a living wiki, where knowledgeable players collectively build and maintain the authoritative dataset.

Governance Principles

- **Ownership with Responsibility:** Users can create and claim initial ownership of entities (e.g., a Quest entry), but are expected to be responsive curators.
- **Community Oversight:** The community has clear pathways to suggest improvements, override stagnation, and ensure data accuracy.
- **Transparency & Auditability:** All changes are tracked, and the provenance of information is always visible.

Core Collaboration Features

1. Quality Signaling (Likable System)

A simple like/dislike (or upvote/downvote) mechanism on all user-submitted entities (Quests, Guide steps, Item details). Provides immediate, crowd-sourced quality indicators and helps surface the most accurate and helpful content.

2. Collaborative Editing via Proposal Requests (PRs)

The primary mechanism for iterative improvement, similar to pull requests in software development.

Workflow:

1. A user spots an inaccuracy or omission in an existing record.
2. They submit a PR with their proposed changes (e.g., correcting a reward item, adding a dialogue step).
3. The original contributor (and/or designated moderators) receives a notification to review the PR.
4. The reviewer can approve or reject the proposal.
5. Upon approval, changes are merged, the record is updated and the action is recorded into the Entity's version history.

3. Content Forking for Community Resolution

Addresses scenarios where an original contributor is unresponsive or refuses valid improvements by allowing users to create a community-owned alternative version.

- **Mechanism:** The forked entity uses a similar naming scheme (e.g., `[Quest Name] (Community Fork)`). The system tracks the fork lineage.
- **Resolution:** Community voting and usage metrics determine which version is presented as primary or "community-verified."

4. Comprehensive Version History & Audit Trail

Every entity maintains a complete version history, capturing:

- **Who:** The user who made the change.
- **What:** A diff of the content changes.
- **When:** Timestamp of the occurrence.
- **Why:** Linked PR description or fork reason.

This allows for understanding the evolution of data and maintaining transparency for all users.

Integration Across Systems

These governance features will be implemented across the Data Repository and Structured Guides System, ensuring both factual data and strategic content remain accurate, current, and shaped by collective player expertise.

Project Vision

This platform aims to become the definitive, community-powered toolset for AION Classic EU players.

By combining a deeply structured, collaboratively maintained knowledge base with intelligent planning tools and an AI interface, it creates a virtuous cycle: community contributions fuel the AI's accuracy, and the AI makes that collective knowledge effortlessly accessible.

This empowers players of all commitment levels to achieve their in-game objectives efficiently and fosters a sustainable ecosystem of shared expertise.