ArchSense - Project Proposal

Project Idea and Motivation

Modern engineering organizations increasingly rely on Jira as the primary system for tracking work across teams, services, and domains. While Jira issues are effective for coordinating tasks and capturing functional requirements, they often lack the architectural context necessary for sound technical decision making. Issues typically describe what needs to be done, but not how the existing system is structured, which components are involved, or how a proposed change fits into the broader architecture.

In practice, architectural knowledge is fragmented across multiple sources. Confluence pages, Notion documents, design presentations, and historical discussions often contain relevant information, but this knowledge is rarely centralized, consistently updated, or directly linked to the issues that depend on it. As a result, engineers are frequently required to reconstruct an architectural understanding manually, relying on partial documentation and personal experience.

ArchSense is motivated by the need to bridge this gap. The project aims to transform a Jira issue from an isolated task description into a structured architectural problem statement. By surfacing relevant architectural context early, ArchSense supports better reasoning, clearer communication, and more informed design decisions before implementation begins.

The project deliberately focuses on architectural sense-making and decision support. It does not aim to automate implementation or replace human judgment, but rather to enhance the quality of architectural understanding within engineering workflows.

Presumed Solution (High-Level)

ArchSense is designed as a context-aware analytical assistant that processes a Jira issue and produces structured technical artifacts that reflect both the current and the expected future state of the system. These artifacts describe architectural components, dependencies, constraints, and potential resolution strategies associated with the issue.

The solution is based on the combination of three controlled sources of input. The first source is structured data from Jira, including issue descriptions, metadata, and relationships. The second source is organizational knowledge retrieved through an enterprise knowledge hub, such as Pipehub, which aggregates documentation and technical context from existing sources. The third source is language model based reasoning, constrained to analysis, synthesis, and explanation rather than free-form generation.

The output of ArchSense is a read-only technical brief. This brief is intended to be consumed by engineers and reviewers as part of design discussions, architectural reviews, or onboarding activities. The system does not modify external systems or execute actions, ensuring that its role remains advisory and analytical.

Problem Being Solved

The primary problem addressed by ArchSense is the absence of explicit architectural understanding at the point where engineering work begins. Jira issues often provide insufficient context to understand the architectural impact of a change, while relevant system knowledge is distributed, incomplete, or outdated.

As a consequence, architectural reasoning is often delayed until implementation is already underway. This leads to design misalignment, overlooked dependencies, increased rework, and inefficient review cycles. New team members and engineers working across domains are particularly affected, as they lack the historical and contextual knowledge required to reason confidently about system changes.

Existing AI-assisted tools largely focus on productivity at the task or code level. They assist with implementation details, text generation, or search, but they do not provide a coherent architectural view that connects intent, structure, and impact.

ArchSense reframes the problem by emphasizing architectural understanding as a first-class concern. Instead of immediately focusing on how to implement a change, the system encourages teams to first establish a shared understanding of the system being modified and the implications of the change.

Expected User Persona

The primary users of ArchSense are software engineers working on medium to large scale systems. These engineers are typically assigned Jira issues that involve architectural considerations such as service boundaries, data flows, or cross-team dependencies. They participate in design discussions and benefit from having architectural context presented clearly and consistently.

Secondary users include architecture engineers, platform engineers, technical leads, and engineering managers. For these users, ArchSense serves as a tool for reviewing proposed changes, validating assumptions, and aligning understanding across teams. They are primarily consumers of the generated artifacts rather than direct operators of the system.

The system assumes a technically proficient audience. However, it does not assume deep familiarity with every subsystem involved, making it particularly useful in cross-domain or cross-team scenarios.

Assumptions

ArchSense is built on several foundational assumptions. Jira functions as the authoritative system of record for engineering work. Organizational knowledge already exists within the company, even if it is fragmented or inconsistently maintained. The system operates strictly in read-only mode and does not perform any autonomous actions.

All outputs are intended for human consumption and review. The success of the system is evaluated based on clarity, correctness, and usefulness of the generated artifacts, rather than operational efficiency or automation.

These assumptions intentionally constrain the scope of the project. They ensure that the system remains feasible within a workshop setting and that its behavior can be evaluated in a controlled and transparent manner.

Alternative Approaches

Several alternative approaches could be used to address the challenges ArchSense targets. Manual documentation practices are common, but they are time-consuming and difficult to keep synchronized with ongoing development. Static architectural documents often become outdated and disconnected from day-to-day work.

Codebase analysis tools provide insight into implementation details, but they do not capture architectural intent or planned future changes. General-purpose AI assistants can generate explanations or summaries, but they often lack structured grounding in the organization’s architecture. Knowledge search tools retrieve relevant documents, but they leave synthesis and interpretation to the user.

ArchSense differentiates itself by combining issue intent with architectural reasoning. Rather than replacing documentation or implementation tools, it acts as a synthesis layer that connects existing knowledge to active engineering work.

Main Features and User Flow

The main features of ArchSense include reconstruction of the current system architecture based on available context, interpretation of the expected future architecture implied by the issue, identification of affected components and dependencies, and explicit articulation of assumptions and constraints. The system also presents multiple resolution strategies, each accompanied by risks and tradeoffs.

All outputs are delivered in the form of a structured technical brief that can be reviewed, shared, and discussed.

A typical user flow begins when an engineer is assigned a Jira issue. ArchSense analyzes the issue, retrieves relevant organizational knowledge, synthesizes an architectural understanding, and generates structured artifacts. These artifacts support design discussions and architectural reviews before implementation begins.

External Dependencies

ArchSense relies on Jira for issue metadata and descriptions, an enterprise knowledge hub such as Pipehub for documentation retrieval, and a large language model provider such as OpenAI or Claude. The system does not require direct integration with code repositories, build systems, or deployment pipelines.
