S8

Software Designs At Labman

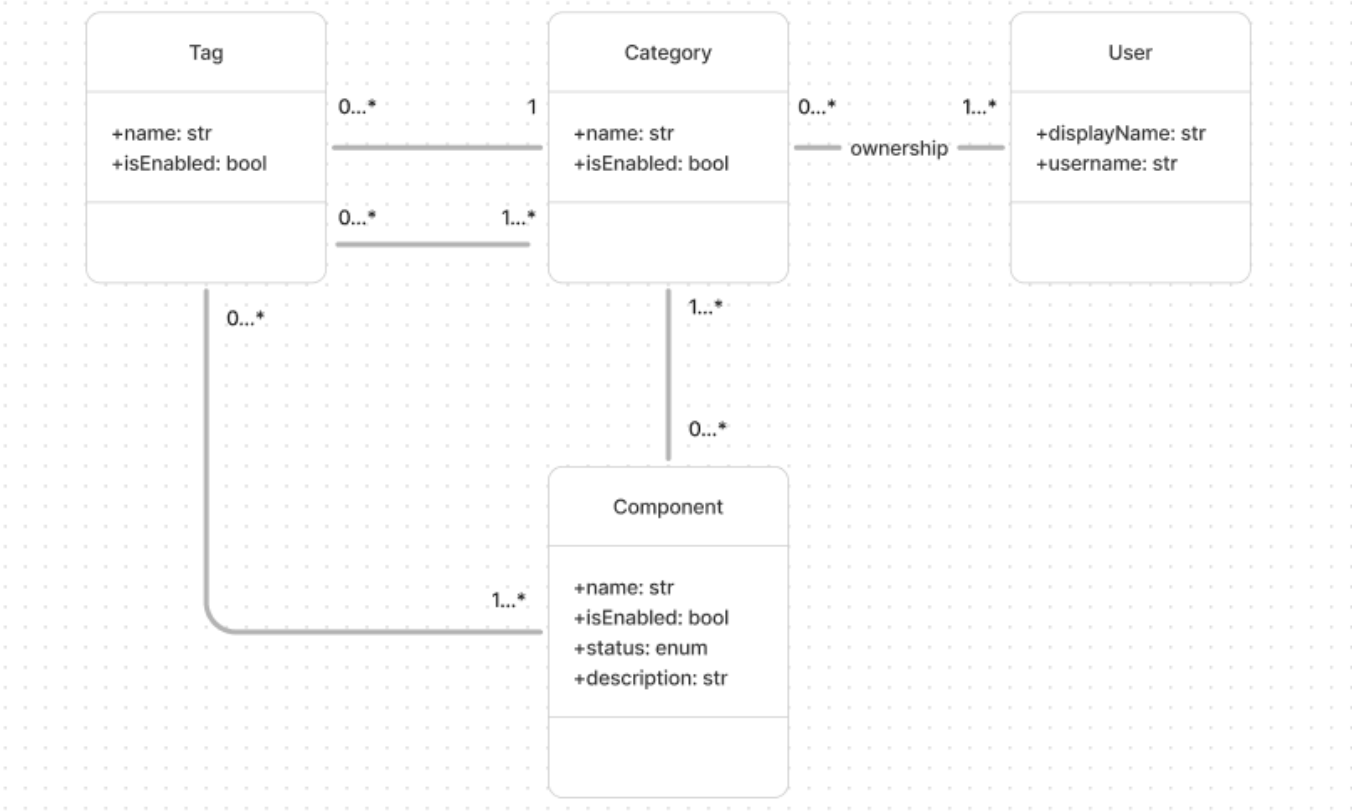
Arthur Clarkson

1st October 2024

At Labman, when designing complex systems, we often draw UML diagrams to aid in understanding. In a recent project involving the technology reference system, the relationships between different entities were mapped out. Each **Category** can have many **Components**, while **Components** can belong to multiple **Categories**, establishing a many-to-many relationship between these entities. Additionally, **Tags** can be associated with **Categories** and **Components**, where a tag can apply to multiple categories and components, and each category or component can have multiple tags.

Users, represented by the **User** entity in the diagram, have ownership over categories, with the ability to own many categories (1..* relationship between **User** and **Category**). The **Component** entity contains several attributes, including `name`, `isEnabled`, `status` (an enum), and `description`, which further defines the characteristics of each component. Similarly, the **Category** and **Tag** entities also have `name` and `isEnabled` attributes to describe their basic information and status within the system.

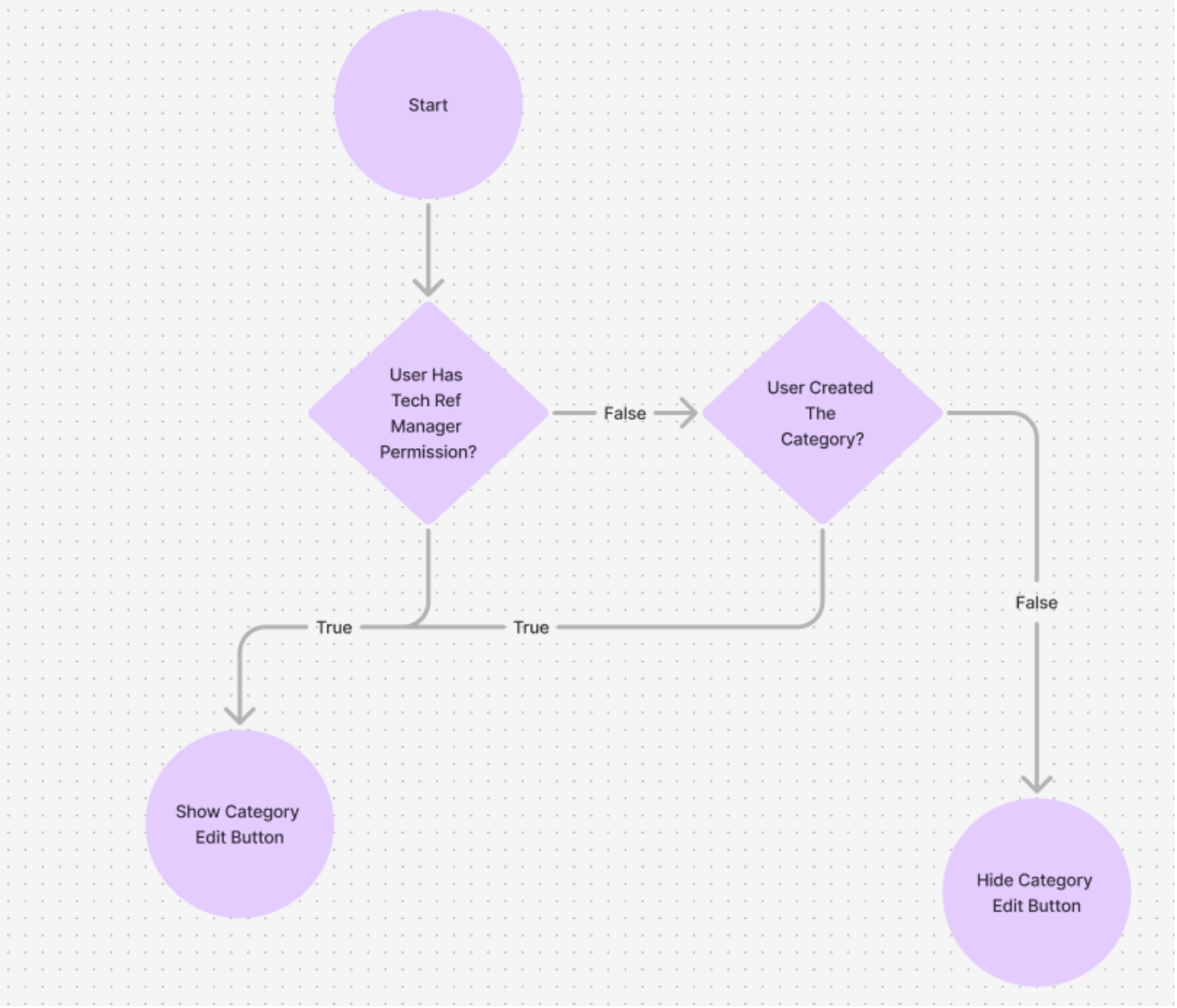
This UML diagram provides a clear visualization of the system's structure and interactions, showing the relationships and constraints between categories, components, tags, and users within the technology reference system.



At Labman, along with UML diagrams, we also heavily rely on flowcharts to map out the workflows and processes in our systems. Flowcharts help us visualize the step-by-step sequence of operations or actions taken within various components of the system. When working on the technology reference system, for example, we use flowcharts to model the lifecycle of a **Component**, from its creation, through approval stages, to when it's tagged, categorized, and made available for use. This provides a clear view of the decision points and potential branches that can occur based on the state of a **Component**, such as whether it’s enabled or disabled, or how its status changes.

Flowcharts complement UML diagrams by illustrating the flow of information or control between different entities, such as how **Users** interact with **Categories** or **Components**, and how **Tags** can influence certain actions. While the UML diagram provides a structural overview, flowcharts dive deeper into how these elements operate over time, providing more clarity on operational workflows, such as how a **User** might create a new **Category**, assign components to it, or update tags.

Here is an example of a flowchart I used, to check whether we show the category edit button on the category page:



By using both UML diagrams and flowcharts in tandem, we create a more comprehensive picture of both the static architecture and dynamic processes within our systems, ensuring a robust and well-understood design. This approach allows us to troubleshoot potential issues early in development and ensure smoother collaboration among team members.