**ATAM:** **Task Management System**

**Project Context:** Create an online platform that enables users to efficiently organize, manage, and track their tasks and projects. The application should include features such as task creation, assignment to team members, setting due dates, priorities, and categories, as well as generating progress and completion reports for tasks.

1. **Definition of Quality Objectives**: We identified the following quality objectives:
   * **Security:** Tasks should only be displayed and protected for each user, so that the user who logs into the platform can exclusively view their created tasks.
   * **Performance:** The visualization and updating of task statuses should be fast even during high-load situations.
   * **Usability:** Tasks should be easily managed within the platform.
2. **Identification of Scenarios:** We defined specific scenarios to evaluate each objective:
   * **Security Scenario:** Prevention of unauthorized access to tasks.
   * **Performance Scenario:** Quick task visualization during peak hours.
   * **Usability Scenario:** Intuitive task visualization and management according to their corresponding status.
3. **Creation of Initial Architectural Model:** We developed an initial architectural model that includes components for task visualization, creation, updating, and deletion.

**Iteration 1:**

So far, we have only had one iteration, but we believe that this single interaction comprehensively reflects the activities that a user can perform in the system.

Diagrama

Descripción generada automáticamenteDiagrama

Descripción generada automáticamente

**Description of Components:**

* **Frontend UI Container**: This component provides the user interface for the application. It includes panels, buttons, and other controls that users interact with to manage their tasks.
* **Backend API Container**: This component provides the application's logic. It includes the code that handles tasks such as creating, updating, deleting, and retrieving tasks.
* **Memory Database**: This component stores application data in memory, such as tasks and users.

**Interactions:**

* + - **El Frontend UI Container** It calls the services in the Backend API Container to perform operations such as creating, updating, deleting, and retrieving tasks.
    - **El Backend API Container** It reads and writes data from the database to perform the operations requested by the Frontend UI Container.

**4. Preliminary Analysis:**

* We evaluate how the architecture addresses each scenario. For example, in the security scenario, we examine how access controls for the user are handled, ensuring that the user can only see the tasks they are managing. Similarly, in the performance scenario, we analyze whether the user can view their tasks without any inconsistencies, regardless of the state in which they are placing them.

**5. Identification of Trade-offs:** We designed the application to be user-friendly and enjoyable, with the primary feature of allowing user to create and adjust their tasks by dragging them to different states as desired.

We discovered that to improve the system's performance, it goes hand in hand with usability. If the system can efficiently manage the state of tasks, performance will be better reflected. We identified a trade-off between performance and usability.

**6. Iteration and Refinement:** We made adjustments to the design to address the identified trade-offs. Initially, we had planned to change the task state with a button, but this was not user-friendly and functional. Therefore, we implemented the drag and drop technique commonly used in UX designs to make the application more intuitive and improve its usability.

**7. Results:**

The detailed analysis of the architecture of the Task Management System has provided us with valuable insights into how our architectural decisions impact the previously defined quality objectives.

We have successfully identified that the application meets the quality objectives, along with the trade-offs and the corresponding refinements, to make the task management system as intuitive and user-friendly as possible for the end user.

**QAW: Task Management System**

**Project Context:** Continuing with the same task management system, we will now apply the Quality Attribute Workshop (QAW) to identify and prioritize quality attributes in collaboration with some familiar individuals willing to provide input on the system.

**1. Identification of Quality Attributes:** In a joint meeting, we gathered stakeholders, including acquaintances, family members, and system developers, to identify quality attributes. Some of the identified attributes include:

* **Functionality**
* **Usability**
* **Reliability**
* **Performance**
* **Security**

**2. Definition of Quality Scenarios:** With the participation of stakeholders, we defined detailed scenarios for each attribute:

* + **Functionality**: The application should allow users to create tasks with a description in the corresponding state, with the option to move the task to any other state and delete them.
  + **Usability**: The application should provide a clear and concise user interface.
  + **Reliability**: The application should not fail when creating, updating, deleting, or retrieving tasks.
  + **Performance**: The application should be able to create, update, delete, or retrieve tasks instantly without any delays.
  + **Security**: The application should only display tasks that belong to the user who logged into the system.

**3. Prioritization of Quality Attributes:**

Using voting techniques and collaborative discussions, stakeholders prioritize the quality attributes. We discovered that functionality, usability, reliability, and performance are critical for the success of the product. Security is also important but has a lower priority.

1. **Definition of Metrics:** For each quality attribute, we defined specific metrics that will help us evaluate the system's performance in those aspects. For example:

* **Functionality**, we could measure the time it takes to create tasks and assign them to their respective states.
* **Usability**, we could measure the average time to complete common tasks.
* **Reliability**, we could consider the reasons why the application might fail when executing any of the task management functions.
* **Performance**, we can evaluate the response time the application takes to validate functions such as task viewing, creation, modification, and deletion.
* **Security**, we can assess the ways in which the user may be vulnerable when their tasks are in unauthorized hands.

**5. Detailed Scenario Development:** In additional sessions, we worked on developing detailed scenarios to better understand how the quality attributes would manifest in specific situations.

**Functionality Scenario: Task Creation and State Change**

* **Description:** A registered user logs into the system and wants to create a new task. The user provides a detailed description of the task and selects a specific state. Then, the user attempts to change the task's state.
* **Expected Result:** The task is successfully created, and the state change occurs instantly. There are no errors or delays in these actions.

**Usability Scenario: Intuitive User Interface**

* **Description:** A user with no prior experience using the system attempts to perform a series of common tasks, such as creating a task or marking a task as completed.
* **Expected Result:** The user can complete these actions intuitively and without confusion. The user interface is clear and easy to navigate, allowing users to efficiently perform these tasks.

**Reliability Scenario: Error-Free Task Management**

* **Description:** The user concurrently uses the system to continuously create, update, delete, and retrieve tasks during a period of high load.
* **Expected Result:** The system demonstrates high reliability in executing these operations, with no errors or system crashes, even under heavy load.

**Performance Scenario: Homepage Loading**

* **Description:** A user opens the web application of the task management system and logs in. The user expects the system's homepage to load quickly and without noticeable delays.
* **Expected Result:** The system's homepage should load quickly and efficiently after the user logs in, providing immediate access to tasks and core functionality. There should be no significant delays in page loading.

**Security Scenario: Restricting Access to Private Data**

* **Description:** A user is authenticated in the system and wants to access the details of a task that does not belong to them. They attempt to directly access another user's task URL.
* **Expected Result:** The system should verify the user's credentials and ensure that they only have access to their own data and tasks. It should deny access to another user's task and display an error message or redirect the user to the homepage.

**Results:**

We have conducted a detailed Quality Attribute Workshop (QAW) for the Task Management System, working collaboratively with various stakeholders, including acquaintances, family members, and system developers. This process has allowed us to make considerable progress in understanding and improving the system's quality, achieving the following key results:

1. **Identified and Prioritized Quality Attributes:** We have identified and prioritized a list of quality attributes critical to the success of the system. These attributes include Functionality, Usability, Reliability, Performance, and Security. This prioritization has provided us with a clear understanding of the areas to focus on in our development and evaluation efforts.
2. **Defined Detailed Scenarios:** For each of these quality attributes, we have defined detailed scenarios that describe specific situations of system use. These scenarios are essential for understanding how quality attributes manifest in practice and will serve as a solid foundation for subsequent testing and evaluations.
3. **Established Metrics for Evaluation:** We have defined specific metrics for each quality attribute to help us evaluate and measure the system's performance in these dimensions. These metrics will provide us with quantitative data to determine if the system meets the established quality standards.
4. **Developed Detailed Scenarios:** Additional detailed scenarios have been developed to better understand how quality attributes manifest in specific usage situations. These scenarios provide a concrete description of user interactions with the system and will help us assess its performance more accurately.