**1. Requirement Gathering & Analysis**

**Goal:**  
The goal of this project is to develop a dynamic and user-friendly **Image Gallery with Filter Options**. This application will allow users to view a collection of images and filter them based on predefined categories or tags such as Nature, Animals, People, and Abstract. The main objective is to enhance user experience by making it easier to browse specific types of images with minimal effort.

**Functional Requirements:**

1. **Image Display**

* All images should be displayed in a clean, organized grid or card layout.
* The layout must be responsive to different screen sizes (desktop, tablet, mobile).

1. **Filtering Feature**

* Users should be able to filter images by categories such as Nature, Animals, People, etc.
* Filtering should happen dynamically without requiring the page to reload.

1. **Category Selection Interface**

* Provide category filters as buttons, checkboxes, or a dropdown menu.
* Include an “All” option to reset the gallery and show all images.

1. **Image Details (Optional)**

* Each image can optionally include a title, caption, or tags for additional context.

**Non-Functional Requirements:**

* **Performance**: The application should load images quickly and apply filters with minimal delay.
* **Responsiveness**: The gallery should work seamlessly on all devices and screen sizes.
* **Accessibility**: Use semantic HTML and ensure the interface is accessible via keyboard navigation.
* **Usability**: The interface should be intuitive and easy to use for all types of users.
* **Scalability**: The system should allow easy addition of new images and categories in the future.

**Data Requirements:**

* Images should be stored with relevant metadata such as file path, title, and category/tag.
* If using a backend, images and metadata should be retrieved from a database or server.

**Technology Stack (Tentative):**

* **Frontend**: HTML, CSS, JavaScript (or any frontend framework like React if required).
* **Backend (Optional)**: Node.js, PHP, or Python (if dynamic image loading is required).
* **Database (Optional)**: To store image information and categories (e.g., MySQL, MongoDB).

**Use Case Summary:**

1. The user opens the image gallery page.
2. All available images are displayed in a grid layout.
3. The user selects a category from the filters (e.g., "Nature").
4. Only images tagged with "Nature" are displayed.
5. The user selects “All” to reset the filter and view all images again.