**3. System Design Phase**

The **System Design Phase** focuses on translating the gathered requirements into a clear blueprint for building the **Image Gallery with Filters**. This phase outlines the structural and functional components of the system, defines the user interface, data flow, and how different modules will interact. The design ensures the system is scalable, maintainable, and easy to use.

**1. System Architecture:**  
The system is designed as a **web-based application** consisting of the following main components:

* **Frontend (Client Side):**  
  Developed using HTML, CSS, and JavaScript. Responsible for rendering the image gallery, displaying filter options, and handling user interactions.
* **Backend (Optional – for dynamic image loading):**  
  Can be developed using PHP, Node.js, or Python to fetch images and metadata from a database. Not mandatory if the gallery uses static images.
* **Database (Optional):**  
  Used to store image URLs, titles, categories, and other metadata. MySQL, MongoDB, or Firebase can be used if dynamic content management is needed.

1. **User Interface Design:**  
   The user interface is designed to be clean, responsive, and user-friendly. It includes the following elements:

* **Gallery Layout:**  
  A grid layout displaying all images in equal-sized cards or thumbnails.
* **Filter Section:**  
  Buttons, checkboxes, or a dropdown menu that allows users to filter images by categories such as Nature, Animals, People, etc.
* **Reset Filter Option:**  
  An “All” button that resets the filters and displays the full gallery.
* **Optional Details:**  
  Each image may include a title, caption, or tags for additional context.

**3. Component Design:**

| **Component** | **Description** |
| --- | --- |
| **Image Container** | Holds all image thumbnails and updates dynamically based on the selected filter. |
| **Filter Menu** | Contains category options for filtering the images. |
| **Image Data** | Stores image file path, category, and metadata (static or dynamic source). |
| **Event Handlers** | Handle user clicks on filters and dynamically update the gallery. |

**4. Data Flow Design:**

* On page load, all images are displayed.
* When a user selects a filter, an event is triggered.
* The system checks each image's category.
* Only images matching the selected category are shown.
* When the user clicks "All", the gallery resets and displays all images.

1. **Tools and Technologies:**

* **Frontend:** HTML5, CSS3, JavaScript (Vanilla JS or libraries like jQuery/React).
* **Backend (Optional):** PHP, Node.js, or Python Flask (for dynamic data).
* **Database (Optional):** MySQL, MongoDB, or Firebase.
* **Design Tools (Optional):** Figma, Adobe XD, or wireframe tools for UI planning.

1. **Responsiveness & Accessibility:**

* Media queries will be used to adapt the layout for different screen sizes.
* Semantic HTML and ARIA labels will be used to improve accessibility.
* Keyboard navigation and proper tab indexing will be ensured.

**7. Security Considerations (if backend is used):**

* Input validation to avoid XSS or injection attacks.
* Proper authentication if admin functionality is included for uploading images.

This design phase ensures that all parts of the **Image Gallery with Filters** are well-structured and organized for smooth development, easy maintenance, and enhanced user experience.