Assignment Overview

Welcome to the frontend assignment for Pixisphere, a platform that connects customers with the best photographers and studios for maternity, newborn, birthday, and other special shoots.

What You'll Build:

You'll be building two core pages that represent real parts of our product:

1. **Category Listing Page** – shows a list of photographers based on a category (e.g., Maternity Photographers in Bengaluru) with filters and sorting options.

2. **Photographer Profile Page** – shows full details of a photographer when a user clicks "View Profile."

These two pages are critical to our customer experience. They help users browse, compare, and choose the right photographer based on preferences like style, rating, and price

Frontend Developer Assignment

Objective:

Build a **Category Listing Page** and a **Photographer Profile Page** using a mock API. This assignment tests your skills in UI development, API fetching, state management, filtering, and logic implementation.

Part 1: Category Listing Page

Example Page:

> Maternity Photographers in Bengaluru

What to Build:

1. Photographer Cards (Grid View)

Each card must display:

- Photographer Name
- Profile Picture
- Location
- Starting Price
- Rating
- Tags (e.g., “Candid”, “Outdoor”)
- "View Profile" button

---

2. Advanced Filters (Sidebar or Drawer)

- **Price Range Slider**

- **Ratings** (e.g., 4+, 3+, etc.)
- **Styles** (checkboxes: Traditional, Candid, Studio, Outdoor, etc.)
- **City Dropdown**
- **Sorting Options:**
    - Price: Low to High
    - Rating: High to Low
    - Recently Added (you can simulate using ID or timestamp)

Filters must update the results dynamically.

---

3. Search Bar (Top)

- Supports searching by name, location, or tag
- Simulate **fuzzy search behavior**

---

4. Pagination or Infinite Scroll

- Load more photographers via infinite scroll or a “Load More” button

---

5. Bonus (Optional)

- AI “Smart Suggestion” strip (e.g., “Top-rated outdoor maternity photographers in Bengaluru”)
- Skeleton loader while fetching data

Part 2: Photographer Profile Page

When a user clicks **“View Profile”**, show:

- Name & Bio
- Styles & Tags
- Price
- Full Gallery (carousel/grid)
- Reviews (Name, Rating, Comment, Date)
- “Send Inquiry” Button (open a form in a modal/popup)

How to Fetch Data

You can use a mock REST API powered by JSON Server.

Step-by-Step:

1. Save the JSON provided below as **`db.json`**
2. Install JSON Server:

bash
CopyEdit
```
npm install -g json-server
```

3. Start the server:

bash
CopyEdit
```
json-server --watch db.json --port 3001
```

4. API Endpoint:

bash
CopyEdit
```
GET http://localhost:3001/photographers
```

API JSON (Mock Data)

json
CopyEdit

```
{
  "photographers": [
    {
      "id": 1,
      "name": "Ravi Studio",
      "location": "Bengaluru",
      "price": 10000,
      "rating": 4.6,
      "styles": ["Outdoor", "Studio"],
      "tags": ["Candid", "Maternity"],
      "bio": "Award-winning studio specializing in maternity and
newborn shoots.",
      "profilePic": "/images/ravi.jpg",
      "portfolio": ["/images/portfolio1.jpg",
"/images/portfolio2.jpg"],
      "reviews": [
        {
          "name": "Ananya",
          "rating": 5,
          "comment": "Truly amazing photos and experience!",
          "date": "2024-12-15"
        }
      ]
    },
    {
      "id": 2,
      "name": "Lens Queen Photography",
```

```
      "location": "Delhi",
      "price": 15000,
      "rating": 4.2,
      "styles": ["Candid", "Indoor"],
      "tags": ["Newborn", "Birthday"],
      "bio": "Delhi-based candid specialist for kids and birthday
parties.",
      "profilePic": "/images/lensqueen.jpg",
      "portfolio": ["/images/lens1.jpg", "/images/lens2.jpg"],
      "reviews": [
        {
          "name": "Priya",
          "rating": 4,
          "comment": "Very professional and punctual!",
          "date": "2024-10-01"
        }
      ]
    },
    {
      "id": 3,
      "name": "Click Factory",
      "location": "Mumbai",
      "price": 8000,
      "rating": 4.8,
      "styles": ["Studio", "Outdoor", "Traditional"],
      "tags": ["Wedding", "Pre-wedding"],
      "bio": "Capturing timeless wedding stories across India.",
      "profilePic": "/images/clickfactory.jpg",
      "portfolio": ["/images/wed1.jpg", "/images/wed2.jpg"],
      "reviews": [
        {
          "name": "Rahul",
          "rating": 5,
          "comment": "We loved every single moment they captured.",
          "date": "2025-01-22"
        }
      ]
    },
```

```
    {
      "id": 4,
      "name": "Moments by Neha",
      "location": "Bengaluru",
      "price": 12000,
      "rating": 4.3,
      "styles": ["Outdoor", "Candid"],
      "tags": ["Maternity", "Couple"],
      "bio": "Natural light specialist focusing on emotional
storytelling.",
      "profilePic": "/images/neha.jpg",
      "portfolio": ["/images/neha1.jpg", "/images/neha2.jpg"],
      "reviews": [
        {
          "name": "Sneha",
          "rating": 4.5,
          "comment": "Captured our maternity journey so beautifully.",
          "date": "2024-11-05"
        }
      ]
    },
    {
      "id": 5,
      "name": "Snapshot Studio",
      "location": "Hyderabad",
      "price": 7000,
      "rating": 3.9,
      "styles": ["Studio"],
      "tags": ["Birthday", "Family"],
      "bio": "Affordable indoor shoots with creative themes.",
      "profilePic": "/images/snapshot.jpg",
      "portfolio": ["/images/snapshot1.jpg", "/images/snapshot2.jpg"],
      "reviews": [
        {
          "name": "Vikram",
          "rating": 3.5,
          "comment": "Decent service, could improve on punctuality.",
          "date": "2024-09-10"
```

```
          }
        ]
      }
    ]
}
```

Technology Requirements

- **Framework**: React with Next.js
- **State Management**: Context API, Redux, or Zustand
- **Styling**: Tailwind CSS, Material UI, or custom CSS
- **Other Expectations**:
    - Debounced search/filter
    - Clean and responsive UI
    - Mobile-first best practices

Deliverables

- GitHub repo with clean commit history
- Live demo (e.g., Netlify, Vercel)
- README including:
    - Setup instructions
    - Short notes on filtering, debounce, and logic
    - Screenshots (optional)

Evaluation Criteria

Area	Details
UI/UX	Responsive, intuitive, modern design
Code Quality	Clear, modular, well-structured components
Filtering Logic	Clean, fast, and user-friendly filtering/sorting/search
API Handling	Proper API calls, state updates, loading states
Performance	Debouncing, memoization, optimized rendering
