நான் முதல்வன்

9530

ST.MOTHER THERESA ENGINEERING COLLEGE

COMPUTER SCIENCE AND ENGINEERING

NM-ID :A78613F6FA3B61CB9FF4FF595B85F5E0

REG.NO:953023104078

Date:22-09-2025

Completed the project named as

Phase-2

SOLUTION DESIGN & ARCHITECTURE

SUBMITTED BY,

MUTHUMARI.K

PH NO:7904776983

Phase 2 – Solution Design & Architecture

1. Tech Stack Selection

Frontend

Framework: React.js (component-based, reusable, fast rendering with Virtual DOM).

Styling: Tailwind CSS (utility-first, responsive classes, easy customization).

Animation: Framer Motion / CSS transitions (for smooth slide effects).

Backend (Optional)

Node.js + Express.js (to create REST APIs for image management).

Storage

Local: JSON file with image paths.

Cloud: Firebase Storage or AWS S3 for scalable storage.

Database (if needed): MongoDB / MySQL for storing image metadata (URLs, captions, order).

2. UI Structure / API Schema Design

UI Components

1. SliderContainer – Main component, handles fetching images, autoplay logic, state management.

2. Slide – Renders an image and optional caption.

3. Navigation – Prev/Next arrows for manual navigation.

4. PaginationDots – Shows the active slide position and allows direct navigation.

5. Loader (optional) – Shows loading state until images are fetched.

API Endpoints (if backend used)

GET /api/images → Fetch all image URLs + metadata.

POST /api/images → Upload a new image.

DELETE /api/images/:id → Delete an image.

PUT /api/images/:id → Update metadata (e.g., caption).

Sample JSON Response:

```
[
    { "id": 1, "url": "https://cdn.com/slider1.jpg", "caption": "Welcome Banner" },
    { "id": 2, "url": "https://cdn.com/slider2.jpg", "caption": "New Collection" }
]
```

3. Data Handling Approach

Image Fetching: Dynamically fetch from API/JSON file.

Autoplay: Use setInterval() in React or custom hooks for automatic slide changes.

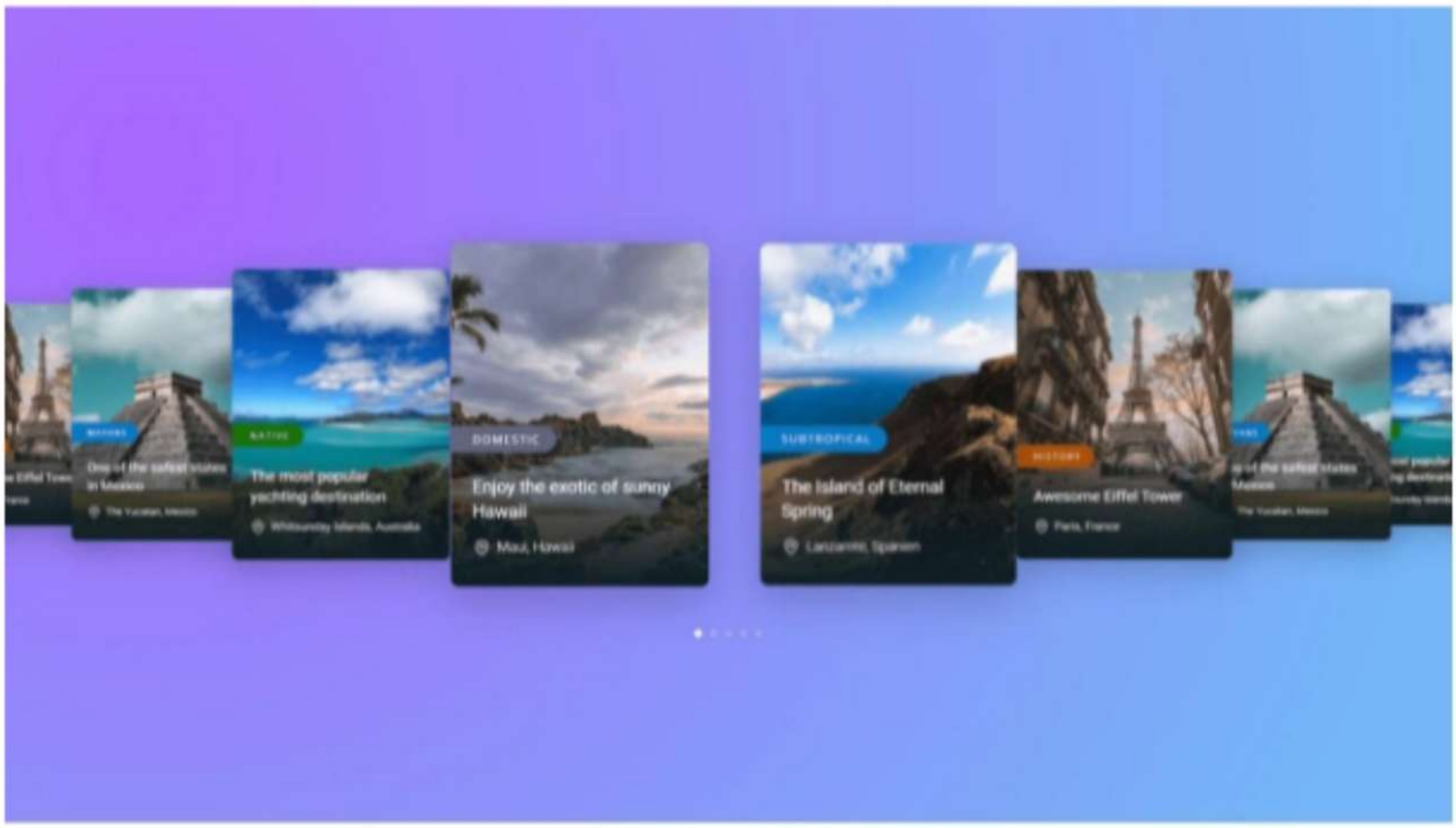
Lazy Loading: Load only current + next/prev images to optimize performance.

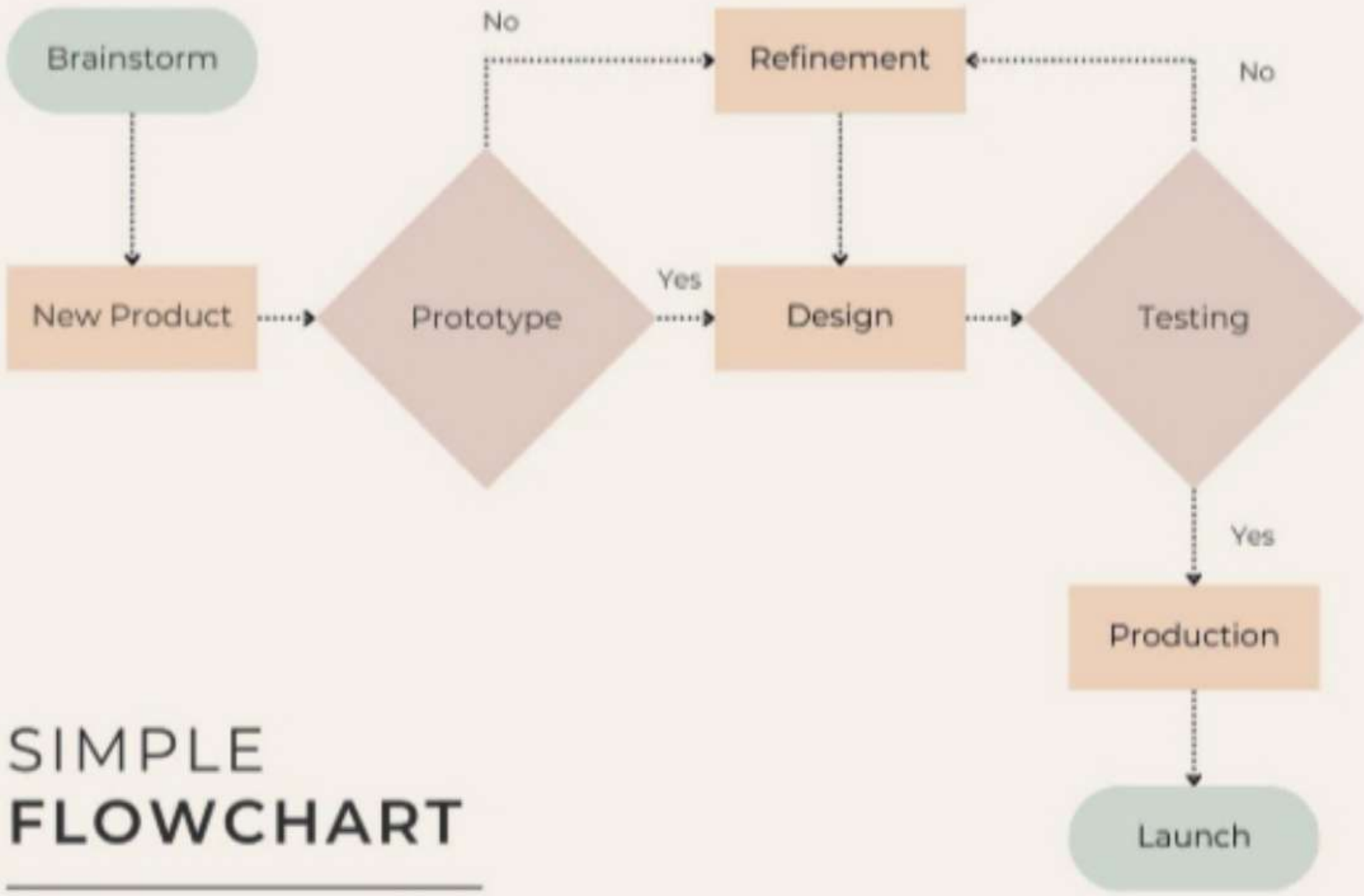
Responsive Design: Ensure compatibility with desktop, tablet, and mobile.

Caching: Use browser cache / service workers for faster reloads.

Error Handling: Fallback image if URL is broken.

4. Component / Module Diagram





SIMPLE **FLOWCHART**