**Installation Steps**  
npm create vite@latest my-social-app -- --template react

**git command:** <https://github.com/thaiGO2003/Advanced-Mobile-Programming_FinalLab>  
git init

git config --global user.email "<email>" git config --global user.name "<name>" git commit -m “<your commit>” git log

cd Advanced-Mobile-Programming\_FinalLab

npm install

npm install react-router-dom axios react-icons

**npm start**  
src/main.jsx  
  
import React from 'react';

import ReactDOM from 'react-dom/client';

import { createBrowserRouter, RouterProvider } from 'react-router-dom';

import StartScreen from './components/StartScreen';

import DetailScreen from './components/DetailScreen';

import './index.css';

const router = createBrowserRouter([

  { path: '/', element: <StartScreen /> },

  { path: '/detail', element: <DetailScreen /> },

]);

ReactDOM.createRoot(document.getElementById('root')).render(

  <React.StrictMode>

    <RouterProvider router={router} />

  </React.StrictMode>,

);

**src/components/StartScreen.jsx**  
import { useNavigate } from "react-router-dom";

import "../styles/StartScreen.css";

function StartScreen() {

  const navigate = useNavigate();

  return (

    <div className="premium-container">

      <div className="overlay">

        <div className="logo">🔊</div>

        <h1 className="title">Welcome to<br />Premium</h1>

        <div className="dots">•••</div>

        <button className="start-button" onClick={() => navigate("/detail")}>

          Start listening

        </button>

      </div>

    </div>

  );

}

export default StartScreen;

**src/styles/StartScreen.css**

.premium-container {

  background-image: url("/Image\_301.jpg");

  background-size: cover;

  background-position: center;

  height: 100vh;

  width: 100vw;

  position: relative;

  color: white;

}

.overlay {

  position: absolute;

  top: 0;

  left: 0;

  right: 0;

  bottom: 0;

  backdrop-filter: brightness(0.8);

  display: flex;

  flex-direction: column;

  align-items: center;

  justify-content: center;

}

.logo {

  font-size: 40px;

  margin-bottom: 40px;

  animation: shake 2s infinite; /\* Logo lắc \*/

}

.title {

  font-size: 32px;

  text-align: center;

  font-weight: bold;

  opacity: 0;

  animation: fadeIn 2s ease-in-out forwards; /\* Xuất hiện dần \*/

}

.dots {

  font-size: 24px;

  margin: 20px 0;

}

.start-button {

  background-color: black;

  color: white;

  border: none;

  padding: 16px 32px;

  font-size: 16px;

  border-radius: 25px;

  cursor: pointer;

  transition: transform 0.3s ease;

}

.start-button:hover {

  transform: scale(1.1); /\* Nút phóng to khi hover \*/

}

/\* Keyframes cho các animation \*/

@keyframes fadeIn {

  from {

    opacity: 0;

    transform: translateY(20px);

  }

  to {

    opacity: 1;

    transform: translateY(0);

  }

}

@keyframes shake {

  0% { transform: rotate(0deg); }

  25% { transform: rotate(5deg); }

  50% { transform: rotate(0deg); }

  75% { transform: rotate(-5deg); }

  100% { transform: rotate(0deg); }

}

**src/components/DetailScreen.jsx**

import React, { useEffect, useState } from "react";

import axios from "axios";

import "../styles/DetailScreen.css";

import { FiChevronLeft, FiBell, FiMoreVertical } from "react-icons/fi";

import { useNavigate } from "react-router-dom";

// API URLs

const API\_URL = "https://6758e6bf60576a194d122d5c.mockapi.io/accounts";

const IMAGE\_API = "https://6758e6bf60576a194d122d5c.mockapi.io/images";

function DetailScreen() {

  const navigate = useNavigate();

  const [users, setUsers] = useState([]);

  const [selectedUser, setSelectedUser] = useState(null);

  const [showMore, setShowMore] = useState(false);

  const [activeTab, setActiveTab] = useState("videos");

  const [videoData, setVideoData] = useState([]);

  useEffect(() => {

    fetchUsers();

    fetchVideos();

  }, []);

  const fetchUsers = async () => {

    try {

      const res = await axios.get(API\_URL);

      setUsers(res.data);

      setSelectedUser(res.data[0]);

    } catch (err) {

      console.error("Error fetching users:", err);

    }

  };

  const fetchVideos = async () => {

    try {

      const res = await axios.get(IMAGE\_API);

      setVideoData(res.data);

    } catch (err) {

      console.error("Error fetching videos:", err);

    }

  };

  const handleFollow = async (user) => {

    try {

      const updatedUser = {

        ...user,

        isFollowed: !user.isFollowed,

        followers: user.isFollowed

          ? (parseInt(user.followers.replace("K", "")) - 1) + "K"

          : (parseInt(user.followers.replace("K", "")) + 1) + "K",

      };

      setUsers((prev) => prev.map((u) => (u.id === user.id ? updatedUser : u)));

      if (selectedUser?.id === user.id) {

        setSelectedUser(updatedUser);

      }

    } catch (err) {

      console.error("Failed to follow user:", err);

    }

  };

  const toggleLike = async (id) => {

    const updatedVideos = videoData.map((video) => {

      if (video.id === id) {

        const updated = { ...video, isLiked: !video.isLiked };

        return updated;

      }

      return video;

    });

    setVideoData(updatedVideos);

  };

  const filteredAccounts = users.filter((user) => user.id !== selectedUser?.id);

  const visibleAccounts = showMore ? filteredAccounts : filteredAccounts.slice(0, 3);

  const renderImages = () => {

    const filtered = activeTab === "liked" ? videoData.filter((video) => video.isLiked) : videoData;

    return filtered.map((video) => (

      <div key={video.id} className="video-card">

        <img

          src={video.image}

          alt={video.name}

          onDoubleClick={() => toggleLike(video.id)}

        />

        <div className="video-info">

          <span>{video.views} views • {video.followers} followers</span>

          <span

            className={`heart ${video.isLiked ? "liked" : ""}`}

            onClick={() => toggleLike(video.id)}

          >

            ♥

          </span>

        </div>

      </div>

    ));

  };

  if (!selectedUser) return <div>Loading...</div>;

  return (

    <div className="profile-container">

      <div className="profile-header-bar">

        <button className="icon-btn" onClick={() => navigate(-1)}>

          <FiChevronLeft size={24} />

        </button>

        <div className="right-icons">

          <button className="icon-btn">

            <FiBell size={20} />

          </button>

          <button className="icon-btn">

            <FiMoreVertical size={20} />

          </button>

        </div>

      </div>

      <div className="profile-header">

        <img className="avatar" src={selectedUser.avatar} alt="Avatar" />

        <h2>{selectedUser.name}</h2>

        <p>{selectedUser.bio}</p>

        <div className="stats">

          <div>

            {selectedUser.following}

            <br />Following

          </div>

          <div>

            {selectedUser.followers}

            <br />Followers

          </div>

          <div>

            {selectedUser.likes}

            <br />Likes

          </div>

        </div>

        <div className="buttons">

          <button className="follow" onClick={() => handleFollow(selectedUser)}>

            {selectedUser.isFollowed ? "Unfollow" : "Follow"}

          </button>

          <button className="message">Message</button>

        </div>

      </div>

      <div className="suggested-section">

        <div className="suggested-header">

          <h3>Suggested accounts</h3>

          <button onClick={() => setShowMore(!showMore)}>

            {showMore ? "View less" : "View more"}

          </button>

        </div>

        <div className={`suggested-list ${showMore ? "expanded" : ""}`}>

          {visibleAccounts.map((user) => (

            <div

              className="suggested-user bordered"

              key={user.id}

              onClick={() => setSelectedUser(user)}

            >

              <img src={user.avatar} alt={user.name} />

              <p>{user.name}</p>

              <button

                onClick={(e) => {

                  e.stopPropagation();

                  handleFollow(user);

                }}

              >

                {user.isFollowed ? "Unfollow" : "Follow"}

              </button>

            </div>

          ))}

        </div>

      </div>

      <div className="tabs">

        <button

          className={activeTab === "videos" ? "active" : ""}

          onClick={() => setActiveTab("videos")}

        >

          ▶ Videos

        </button>

        <button

          className={activeTab === "liked" ? "active" : ""}

          onClick={() => setActiveTab("liked")}

        >

          ❤️ Liked

        </button>

      </div>

      <div className="video-grid">{renderImages()}</div>

    </div>

  );

}

**export default DetailScreen;**

src/styles/DetailScreen.css

.profile-container {

  width: 100vw;

  height: 100vh;

  font-family: sans-serif;

  overflow-y: auto; /\* Hỗ trợ cuộn nếu nội dung dài \*/

  padding: 10px; /\* Thêm padding để tránh dính mép \*/

}

.profile-header {

  text-align: center;

  margin-bottom: 30px;

}

.avatar {

  width: 100px;

  height: 100px;

  border-radius: 50%;

  object-fit: cover; /\* Giữ tỷ lệ hình ảnh \*/

}

.stats {

  display: flex;

  justify-content: space-around;

  margin: 10px 0;

  font-weight: bold;

}

.buttons {

  display: flex;

  justify-content: center;

  gap: 10px;

}

.follow,

.message {

  padding: 8px 16px;

  border: none;

  border-radius: 20px;

  cursor: pointer;

}

.follow {

  background-color: #1da1f2;

  color: white;

}

.message {

  background-color: white;

  color: #1da1f2;

  border: 1px solid #1da1f2;

}

.suggested-section {

  margin: 30px 0;

}

.suggested-header {

  display: flex;

  justify-content: space-between;

  align-items: center;

  width: 96vw;

  padding: 10px;

}

.suggested-list {

  display: flex;

  flex-wrap: wrap;

  justify-content: center;

  gap: 10px;

  margin-top: 10px;

}

.suggested-user {

  width: 20vw;

  padding: 10px;

  text-align: center;

  border-radius: 10px;

  cursor: pointer;

  background-color: #f9f9f9;

}

.suggested-user img {

  width: 60px;

  height: 60px;

  border-radius: 50%;

  object-fit: cover;

}

.suggested-user button {

  margin-top: 5px;

  padding: 5px 10px;

  background-color: #1da1f2;

  color: white;

  border: none;

  border-radius: 12px;

  font-size: 12px;

  cursor: pointer;

}

.suggested-user.bordered {

  border: 2px solid #1da1f2;

  transition: transform 0.2s;

}

.suggested-user.bordered:hover {

  transform: scale(1.05);

}

.tabs {

  display: flex;

  justify-content: center;

  gap: 20px;

  margin-top: 20px;

}

.tabs button {

  padding: 10px 20px;

  border: none;

  background-color: #eee;

  border-radius: 20px;

  cursor: pointer;

}

.tabs .active {

  background-color: #1da1f2;

  color: white;

}

.video-grid {

  display: grid;

  grid-template-columns: repeat(2, 1fr); /\* Mặc định 2 cột \*/

  gap: 15px;

  margin-top: 20px;

  padding: 0 10px;

}

@media (min-width: 600px) {

  .video-grid {

    grid-template-columns: repeat(auto-fit, minmax(180px, 1fr));

  }

}

.video-card {

  position: relative;

  border: 1px solid #ddd;

  border-radius: 8px;

  overflow: hidden;

  aspect-ratio: 3 / 4; /\* Giữ tỷ lệ 4:3 \*/

}

.video-card img {

  width: 100%;

  height: 100%;

  object-fit: cover; /\* Đảm bảo hình ảnh lấp đầy mà không méo \*/

}

.video-info {

  position: absolute;

  bottom: 0;

  left: 0;

  width: 100%;

  padding: 10px;

  color: white;

  font-size: 14px;

  display: flex;

  justify-content: space-between;

  align-items: center;

  background: linear-gradient(to top, rgba(0, 0, 0, 0.7), transparent);

  box-sizing: border-box;

}

.heart {

  display: flex;

  justify-content: center;

  align-items: center;

  font-size: 24px; /\* Giảm kích thước để vừa với video-info \*/

  cursor: pointer;

  transition: transform 0.2s;

}

.heart.liked {

  font-size: 24px;

  color: red;

}

.heart:hover {

  transform: scale(1.2);

}

/\* ProfileHeader styles \*/

.profile-header-bar {

  display: flex;

  justify-content: space-between;

  align-items: center;

  padding: 12px 16px;

  background-color: #fff;

  border-bottom: 1px solid #eee;

  position: sticky;

  top: 0;

  z-index: 10;

}

.icon-btn {

  background: none;

  border: none;

  cursor: pointer;

  padding: 4px;

  color: #333;

}

.right-icons {

  display: flex;

  gap: 12px;

}