

Đề thi thực hành React JS

Thời gian làm bài: 14 ngày (2 tuần)

TỪ ĐIỂN THUẬT NGỮ

A-C

- **API (Application Programming Interface):** Giao diện lập trình ứng dụng - cách để frontend "nói chuyện" với backend
- **Authentication:** Xác thực - quá trình kiểm tra người dùng có phải là ai họ nói không
- **Authorization:** Phân quyền - kiểm tra người dùng có quyền làm gì không
- **Breakpoint:** Điểm ngắt - kích thước màn hình để thay đổi layout (mobile, tablet, desktop)
- **CRUD:** Create (Tạo), Read (Đọc), Update (Sửa), Delete (Xóa) - 4 thao tác cơ bản với dữ liệu
- **CSS:** Cascading Style Sheets - ngôn ngữ để trang trí giao diện web
- **Component:** Thành phần - một phần của giao diện có thể tái sử dụng (như button, form, card)

D-H

- **Debounce:** Kỹ thuật trì hoãn thực hiện hành động cho đến khi người dùng ngừng thao tác (VD: search sau khi ngừng gõ 500ms)
- **Deploy/Deployment:** Triển khai - đưa ứng dụng lên internet để mọi người có thể truy cập
- **DOM (Document Object Model):** Mô hình cây của trang web, React dùng để thay đổi nội dung trang
- **Endpoint:** Điểm cuối - URL của API (VD: /api/users là endpoint để lấy danh sách users)
- **Error Handling:** Xử lý lỗi - cách ứng dụng phản ứng khi có lỗi xảy ra
- **Frontend:** Phần giao diện người dùng nhìn thấy và tương tác
- **Hook:** Móc câu - function đặc biệt của React để quản lý state và lifecycle (useState, useEffect)

I-P

- **JSON:** JavaScript Object Notation - định dạng dữ liệu phổ biến, giống object JS
- **JWT (JSON Web Token):** Mã thông báo để xác thực người dùng
- **localStorage:** Kho lưu trữ local của browser, data vẫn còn sau khi tắt browser
- **Middleware:** Phần mềm trung gian - code chạy giữa request và response

- **Mobile-first:** Thiết kế cho mobile trước, sau đó mở rộng ra desktop
- **Props:** Properties - cách truyền dữ liệu từ component cha xuống component con
- **PWA (Progressive Web App):** Ứng dụng web có thể hoạt động như app mobile

Q-Z

- **Query Parameters:** Tham số truy vấn - phần sau dấu ? trong URL (?page=1&limit=10)
- **Repository (Repo):** Kho lưu trữ code trên GitHub
- **Responsive:** Giao diện tự động thích ứng với kích thước màn hình khác nhau
- **REST API:** Kiểu thiết kế API phổ biến với GET, POST, PUT, DELETE
- **Route/Routing:** Định tuyến - thay đổi nội dung trang dựa vào URL
- **Schema:** Cấu trúc dữ liệu - định nghĩa data có những field gì
- **State:** Trạng thái - dữ liệu có thể thay đổi của component
- **Token:** Mã thông báo - chuỗi ký tự dùng để xác thực người dùng
- **UI/UX:** User Interface (Giao diện) / User Experience (Trải nghiệm người dùng)
- **Validation:** Kiểm tra tính hợp lệ của dữ liệu người dùng nhập vào

Thông tin chung

Yêu cầu

- Xây dựng ứng dụng **Task Management System** (Hệ thống quản lý công việc)
- Sử dụng React Hooks (useState, useEffect)
- API backend sử dụng JSON Server
- Responsive design
- Deploy ứng dụng lên Vercel/Netlify

Timeline đề xuất (14 ngày)

Tuần 1 (Ngày 1-7):

- └─ Ngày 1-2: Setup project + JSON Server + Authentication
- └─ Ngày 3-4: Basic CRUD (Read + Create tasks)
- └─ Ngày 5-6: Update + Delete + Form validation
- └─ Ngày 7: Debugging + testing các chức năng

Tuần 2 (Ngày 8-14):

- └─ Ngày 8-10: UI/UX styling + responsive design
- └─ Ngày 11-12: Advance features (chọn 1 trong 4 options)
- └─ Ngày 13: Deployment + environment setup
- └─ Ngày 14: Polish + documentation + submission

Tiêu chí chấm điểm

- **Chức năng cơ bản (60 điểm):** CRUD, Authentication, Form validation
- **UI/UX (20 điểm):** Giao diện đẹp, responsive, user-friendly
- **Code quality (10 điểm):** Clean code, component structure, error handling
- **Phần advance (10 điểm):** Tính năng nâng cao và deployment

Hỗ trợ cho học viên

- **Starter template** được cung cấp với basic structure
- **Office hours** mỗi thứ 3 và thứ 6 để hỗ trợ debugging
- **Sample code** cho các phần khó (authentication, API calls)
- **Video tutorials** cho setup JSON Server và deployment
- **Discord/Slack channel** cho Q&A nhanh giữa các buổi

Nộp bài

- Source code trên GitHub (public repository)
- Link demo app đã deploy
- File README.md với hướng dẫn chạy project
- **Progress report** - ghi lại quá trình làm bài và những khó khăn gặp phải

STARTER TEMPLATE








Download Starter Code

GitHub Repository: [https://github.com/\[instructor\]/react-task-management-starter](https://github.com/[instructor]/react-task-management-starter)

Cấu trúc Starter Template đã cung cấp:






```
task-management-starter/
├── public/
│   └── index.html
├── src/
│   ├── components/
│   │   ├── Auth/
│   │   │   ├── LoginForm.js      // ⚡ Basic structure
│   │   │   └── Navbar.js        // ⚡ Basic structure
│   │   ├── Tasks/
│   │   │   ├── TaskList.js      // ⚡ Basic structure
│   │   │   ├── TaskItem.js      // ⚡ Basic structure
│   │   │   ├── TaskForm.js      // ⚡ Basic structure
│   │   │   └── TaskManagement.js // ⚡ Basic structure
│   │   └── Common/
│   │       ├── LoadingSpinner.js // ✅ Complete
│   │       └── ErrorMessage.js   // ✅ Complete
│   ├── api/
│   │   ├── authAPI.js           // ⚡ Function signatures only
│   │   └── taskAPI.js           // ⚡ Function signatures only
│   ├── styles/
│   │   ├── App.css              // ⚡ Basic layout
│   │   └── components.css       // ⚡ Component starters
│   ├── utils/
│   │   └── validation.js        // ✅ Complete validation functions
│   ├── App.js                  // ⚡ Basic routing structure
│   └── index.js                 // ✅ Complete
├── db.json                     // ✅ Complete with sample data
├── package.json                // ✅ All dependencies included
├── README-TEMPLATE.md          // ✅ Template for final README
└── .env.example                 // ✅ Environment variables template
```

Những gì đã hoàn thành trong Starter:

-  **Project setup** - CRA với tất cả dependencies
-  **Basic file structure** - All components tạo sẵn với basic JSX
-  **Validation functions** - Copy từ tài liệu validation guide
-  **Sample data** - db.json với realistic Vietnamese data
-  **CSS starters** - Basic styling và responsive breakpoints
-  **Common components** - LoadingSpinner, ErrorMessage hoàn chỉnh
-  **API function signatures** - Function names và parameters defined

Những gì học viên cần làm:

-  **Implement business logic** trong từng component

-  **Complete API functions** - fetch, error handling
-  **Add form validation** - integrate validation functions
-  **Complete styling** - make it beautiful và responsive
-  **Add advance features** - choose 1 option
-  **Deploy** - setup production environment

Sample từ Starter Template:

src/components/Auth/LoginForm.js

 (Starter):

javascript

```
import React, { useState } from 'react';
import { login } from '../api/authAPI';

function LoginForm({ onLogin }) {
  const [formData, setFormData] = useState({
    username: '',
    password: ''
  });

  const [formState, setFormState] = useState({
    isSubmitting: false,
    error: null
  });

  // TODO: Implement handleInputChange
  const handleInputChange = (e) => {
    // Your code here
  };

  // TODO: Implement handleSubmit with validation
  const handleSubmit = async (e) => {
    e.preventDefault();
    // Your code here - remember to:
    // 1. Validate form data
    // 2. Call login API
    // 3. Handle success/error
    // 4. Call onLogin callback
  };

  return (
    <div className="login-container">
      <div className="login-form">
        <h2>Đăng nhập</h2>

        { /* TODO: Add error display */ }

        <form onSubmit={handleSubmit}>
          { /* TODO: Add form inputs with proper validation */ }

          <button type="submit" disabled={formState.isSubmitting}>
            {formState.isSubmitting ? 'Đang đăng nhập...' : 'Đăng nhập'}
          </button>
        </form>

        { /* Hint: Demo credentials */ }
        <div className="login-help">
```

```
    <p>Demo: admin / 123456</p>
  </div>
</div>
</div>
);
}
```

```
export default LoginForm;
```

`src/api/authAPI.js` (Starter):

```
javascript

const API_BASE_URL = process.env.REACT_APP_API_URL || 'http://localhost:3001';

// TODO: Implement login function
export const login = async (credentials) => {
  // Hints:
  // 1. Fetch from /auth endpoint
  // 2. Find matching user in auth.users array
  // 3. Return user + fake token if valid
  // 4. Throw error if invalid

  // Your implementation here
};

// TODO: Implement other auth functions if needed
export const logout = () => {
  // Your implementation here
};
```

PHẦN 1: CHỨC NĂNG CƠ BẢN (60 điểm)

1.1. Setup Project và JSON Server (5 điểm)

Yêu cầu:

- ✔ Clone starter template từ GitHub
- 🔧 Install dependencies và chạy được project
- 🔧 Setup JSON Server với db.json có sẵn
- 🔧 Environment configuration - setup .env file

Starter đã cung cấp:

- Project structure hoàn chỉnh

- All dependencies trong package.json
- Sample data trong db.json
- Scripts để chạy dev server

Database schema (db.json):

Schema = Cấu trúc dữ liệu - định nghĩa data có những field gì và kiểu dữ liệu

json


```
```json
{
 "users": [
 {
 "id": 1,
 "username": "admin",
 "password": "123456",
 "email": "admin@taskmanager.com",
 "role": "admin"
 },
 {
 "id": 2,
 "username": "user1",
 "password": "password",
 "email": "user1@taskmanager.com",
 "role": "user"
 }
],
 "tasks": [
 {
 "id": 1,
 "title": "Học React Hooks",
 "description": "Hoàn thành bài học về useState và useEffect",
 "status": "todo",
 "priority": "high",
 "dueDate": "2024-08-15",
 "createdBy": 1,
 "createdAt": "2024-08-01T10:00:00Z"
 },
 {
 "id": 2,
 "title": "Setup JSON Server",
 "description": "Cài đặt và cấu hình JSON Server cho project",
 "status": "in-progress",
 "priority": "medium",
 "dueDate": "2024-08-10",
 "createdBy": 1,
 "createdAt": "2024-08-01T11:00:00Z"
 }
],
 "categories": [
 {
 "id": 1,
 "name": "Development",
 "color": "#3498db"
 }
],
}
```

```
{
 "id": 2,
 "name": "Learning",
 "color": "#e74c3c"
},
{
 "id": 3,
 "name": "Personal",
 "color": "#27ae60"
}
]
```

## 1.2. Authentication System (15 điểm)

### Yêu cầu:

- **LoginForm component** với validation
- **Logout functionality**
- **Protected routes** - Chỉ user đã login mới truy cập được (*Route = đường dẫn URL, Protected = được bảo vệ*)
- **Token management** với localStorage (*Token = mã xác thực, localStorage = lưu trữ local trên browser*)
- **Navbar** hiển thị trạng thái login

### Validation rules cho Login:

- Username: không được trống, ít nhất 3 ký tự
- Password: không được trống, ít nhất 6 ký tự
- Error messages rõ ràng và user-friendly

### UI Requirements:

- Login page với form centered
- Navbar hiển thị "Welcome, [username]" + Logout button
- Redirect về login khi logout
- Loading state khi đang login

## 1.3. Task CRUD Operations (25 điểm)

### Yêu cầu chức năng:

- **Create Task:** Form thêm task mới
- **Read Tasks:** Hiển thị danh sách tasks

- **Update Task:** Chỉnh sửa task
- **Delete Task:** Xóa task với confirmation

**Task Form validation:**

- **Title:** Bắt buộc, 5-100 ký tự
- **Description:** Tùy chọn, tối đa 500 ký tự
- **Status:** Bắt buộc (todo, in-progress, completed)
- **Priority:** Bắt buộc (low, medium, high)
- **Due Date:** Bắt buộc, không được trong quá khứ

**Task Display:**

- **Task List** với các thông tin: title, status, priority, due date
- **Task Item** với buttons: Edit, Delete
- **Empty state** khi chưa có tasks
- **Loading state** khi fetch data

**Components cần tạo:**

```
src/
├── components/
│ ├── Auth/
│ │ ├── LoginForm.js
│ │ └── Navbar.js
│ ├── Tasks/
│ │ ├── TaskList.js
│ │ ├── TaskItem.js
│ │ ├── TaskForm.js
│ │ └── TaskManagement.js
│ └── Common/
│ ├── LoadingSpinner.js
│ └── ErrorMessage.js
├── api/
│ ├── authAPI.js
│ └── taskAPI.js
├── App.js
└── index.js
```

**1.4. Form Validation và Error Handling (15 điểm)**

**Validation Requirements:**

- **Real-time validation** khi user typing

- **Submit validation** trước khi gửi API
- **Error messages** cụ thể cho từng field
- **Field highlighting** khi có lỗi
- **Success feedback** khi submit thành công

**Error Handling:**

- **Network errors** - Hiển thị "Kiểm tra kết nối mạng"
- **HTTP errors** - Hiển thị message phù hợp (404, 500...)
- **Validation errors** - Focus vào field đầu tiên có lỗi
- **Retry functionality** - Button "Thử lại" khi có lỗi

**API Error Handling Pattern:**

```
javascript

try {
 const result = await createTask(taskData);
 // Success handling
} catch (error) {
 if (error.name === 'TypeError') {
 setError('Không thể kết nối đến server');
 } else if (error.message.includes('400')) {
 setError('Dữ liệu không hợp lệ');
 } else {
 setError('Đã xảy ra lỗi: ' + error.message);
 }
}
```

**PHẦN 2: UI/UX DESIGN (20 điểm)**

**2.1. Responsive Design (10 điểm)**

**Yêu cầu:**

- **Mobile-first approach** - Thiết kế cho mobile trước (*Mobile-first = ưu tiên thiết kế cho điện thoại trước*)
- **Breakpoints:** Mobile (< 768px), Tablet (768-1024px), Desktop (> 1024px) (*Breakpoint = điểm ngắt để thay đổi giao diện*)
- **Navigation:** Hamburger menu cho mobile (*Hamburger menu = menu 3 gạch ngang trên mobile*)
- **Task cards:** Stack vertically trên mobile, grid trên desktop

- **Form:** Single column trên mobile, có thể multi-column trên desktop

## 2.2. User Experience (10 điểm)

### UX Requirements:

- **Loading states** - Spinner hoặc skeleton loading
- **Empty states** - Illustration/message khi không có data
- **Success feedback** - Toast notifications hoặc success messages
- **Confirmation dialogs** - Confirm trước khi delete
- **Form UX:** Auto-focus, tab navigation, enter to submit
- **Visual hierarchy** - Typography, spacing, colors phù hợp

### Suggested Color Scheme:

```
CSS

:root {
 --primary: #3498db;
 --success: #27ae60;
 --danger: #e74c3c;
 --warning: #f39c12;
 --info: #9b59b6;
 --light: #ecf0f1;
 --dark: #2c3e50;
}
```

## PHẦN 3: CODE QUALITY (10 điểm)

### 3.1. Component Structure (5 điểm)

#### Yêu cầu:

- **Single Responsibility** - Mỗi component có 1 nhiệm vụ rõ ràng (*Single Responsibility = nguyên tắc mỗi component chỉ làm 1 việc*)
- **Props interface** - Rõ ràng về data flow (*Props = dữ liệu truyền giữa components, Interface = giao diện kết nối*)
- **Reusable components** - Tách common components (*Reusable = có thể tái sử dụng, Common = chung*)
- **Proper naming** - Component names và function names rõ ràng
- **File organization** - Group components theo feature


### 3.2. Code Standards (5 điểm)


Requirements:

- **Consistent formatting** - Indentation, spacing
- **Error boundaries** - Proper try-catch blocks
- **Performance** - Avoid unnecessary re-renders
- **Comments** - Comment cho logic phức tạp
- **Constants** - Extract magic numbers/strings

Example Pattern:

```
javascript

//  Good
const TASK_STATUS = {
 TODO: 'todo',
 IN_PROGRESS: 'in-progress',
 COMPLETED: 'completed'
};

//  Bad
if (task.status === 'todo') { ... }
```

PHẦN 4: ADVANCE FEATURES (10 điểm)

4.1. Chọn 1 trong các tính năng sau (8 điểm):

Option A: Search và Filter System (*Search = tìm kiếm, Filter = lọc*)

- **Search:** Tìm kiếm tasks theo title/description
- **Filter by status:** All, Todo, In Progress, Completed (*Filter by = lọc theo*)
- **Sort:** Theo due date, created date, priority (*Sort = sắp xếp*)
- **Debounced search** - Không search mỗi keystroke (*Debounce = trì hoãn search đến khi ngừng gõ*)

Option B: Drag & Drop Task Management (*Drag & Drop = kéo thả*)

- **Kanban Board:** 3 columns (Todo, In Progress, Completed) (*Kanban = bảng quản lý công việc kiểu Nhật*)
- **Auto-save:** Tự động update status khi drop (*Auto-save = tự động lưu*)
- **Visual feedback:** Highlight drop zones (*Visual feedback = phản hồi trực quan*)

Option C: Advanced Form Features (*Advanced = nâng cao*)

- **Multi-step form:** Chia task form thành nhiều steps (*Multi-step = nhiều bước*)
- **Auto-save draft:** Lưu form data tạm thời (*Draft = bản nháp*)
- **Rich text editor:** WYSIWYG editor cho description (*WYSIWYG = What You See Is What You Get*)

**Option D: Dashboard & Analytics** (*Dashboard = bảng điều khiển, Analytics = phân tích*)

- **Chart visualization:** Pie chart cho task status (*Chart = biểu đồ, Pie chart = biểu đồ tròn*)
- **Progress tracking:** Progress bar cho completion rate (*Progress = tiến độ, Tracking = theo dõi*)
- **Export feature:** Export tasks to CSV/JSON (*Export = xuất dữ liệu ra file*)

**4.2. Deployment (2 điểm)**

Yêu cầu:

- **Deploy frontend** lên Vercel hoặc Netlify (*Deploy = triển khai lên internet*)
- **Deploy JSON Server** lên Vercel hoặc Heroku (*Server = máy chủ backend*)
- **Environment variables** - Proper config cho production (*Environment variables = biến môi trường, Production = môi trường thật*)
- **Custom domain** (bonus) - Setup domain name riêng (*Domain = tên miền như google.com*)

**HƯỚNG DẪN NỘP BÀI**

**1. GitHub Repository**

Cấu trúc repository: (*Repository = kho lưu trữ code trên GitHub*)

```
task-management-app/
├── README.md
├── package.json
├── public/
├── src/
├── db.json
└── screenshots/
 ├── login.png
 ├── dashboard.png
 ├── task-form.png
 └── mobile.png
```

**README.md phải bao gồm:**

markdown



# # Task Management System

## ## Mô tả

Ứng dụng quản lý công việc được xây dựng với React và JSON Server.

## ## Tính năng

- Authentication (Login/Logout)
- CRUD operations cho tasks
- Form validation
- Responsive design
- [Tính năng advance đã chọn]

## ## Công nghệ sử dụng

- React 18
- JSON Server
- CSS3/Styled Components
- [Thư viện khác đã sử dụng]

## ## Cài đặt và chạy

### ### Prerequisites

- Node.js >= 14
- npm hoặc yarn

### ### Installation

```
\\\`bash
git clone [repo-url]
cd task-management-app
npm install
\\\`
```

### ### Chạy ứng dụng

```
\\\`bash
Terminal 1: Chạy JSON Server
npm run server
```

### # Terminal 2: Chạy React app

```
npm start
\\\`
```

## ## Demo

- **\*\*Live demo\*\***: [URL deployed app]
- **\*\*API endpoint\*\***: [URL JSON Server]

## ## Screenshots

[Thêm screenshots của app]

## Tác giả

[Tên học viên]

## 2. Submission Checklist

### Trước khi nộp bài, kiểm tra:

- ☐ Tất cả chức năng CRUD hoạt động
- ☐ Authentication working properly
- ☐ Form validation complete
- ☐ Responsive trên mobile/desktop
- ☐ Error handling proper
- ☐ Code clean và organized
- ☐ Tính năng advance hoạt động
- ☐ App deployed successfully
- ☐ README.md đầy đủ thông tin
- ☐ Screenshots đính kèm

## 3. Bonus Points (Tối đa +5 điểm)

- Dark/Light theme** toggle (+1 điểm) (*Theme = chủ đề màu sắc, Toggle = chuyển đổi*)
- Internationalization** (i18n) - English/Vietnamese (+2 điểm) (*i18n = đa ngôn ngữ*)
- Progressive Web App** (PWA) features (+2 điểm) (*PWA = ứng dụng web như app mobile*)
- Unit tests** với Jest/React Testing Library (+3 điểm) (*Unit test = kiểm thử từng phần nhỏ*)
- Custom animations** với CSS/Framer Motion (+1 điểm)

## TIÊU CHÍ CHẤM CHI TIẾT

### Excellent (9-10 điểm mỗi phần):

- Hoàn thành đầy đủ requirements
- Code quality cao, best practices
- UI/UX professional level
- Advance features impressive
- Perfect deployment

### Good (7-8 điểm mỗi phần):

- Hoàn thành hầu hết requirements
- Code clean, ít bugs
- UI/UX tốt, responsive

- Advance features working
- Deployment successful

**Average (5-6 điểm mỗi phần):**

- Hoàn thành basic requirements
- Code có một số issues
- UI cơ bản, chưa responsive
- Advance features basic
- Deployment có issues

**Below Average (0-4 điểm mỗi phần):**

- Chưa hoàn thành requirements
- Code có nhiều bugs
- UI poor, không responsive
- Advance features không work
- Deployment failed

---

**LƯU Ý QUAN TRỌNG**




1. **Không copy-paste code** từ internet mà không hiểu
2. **Commit thường xuyên** để show progress
3. **Test thoroughly** trước khi nộp bài
4. **Document code** khi cần thiết
5. **Ask questions** nếu không rõ requirements









**Deadline: [Ngày tháng năm] 23:59 (14 ngày từ khi nhận đề)**

**OFFICE HOURS SCHEDULE**

- **Thứ 3, 19:00-20:00:** Q&A chung, giải đáp thắc mắc
- **Thứ 6, 19:00-20:00:** Code review, debugging support
- **Discord/Slack:** Hỗ trợ nhanh 24/7

**SUBMISSION CHECKLIST (Kiểm tra trước khi nộp)**

- ☐ ☒ Clone được starter template
- ☐  Tất cả chức năng CRUD hoạt động
- ☐  Authentication working properly
- ☐  Form validation complete

- ☐  Responsive trên mobile/desktop
- ☐  Error handling proper
- ☐  Code clean và organized
- ☐  Advance feature hoạt động (1 trong 4)
- ☐  App deployed successfully
- ☐  README.md đầy đủ thông tin
- ☐  Progress report written
- ☐  Screenshots đính kèm

**Chúc các bạn làm bài tốt!** 

Lưu ý:  = Đã có sẵn trong starter,  = Cần implement,  = Cần viết documentation