# Assignment Reference Document

## Introduction

### Overview

This document provides a comprehensive guide for completing a ReactJS-based frontend developer internship assignment. The objective is to design and implement an admin dashboard for a social media application. The dashboard will allow administrators to manage users and posts effectively.

### Purpose and Goals

1. **Practice and demonstrate ReactJS and Next.js skills**.

2. **Develop a functional, user-friendly admin dashboard**.

3. **Show capability in handling dummy data and authentication process**.

## Step-by-Step Instructions

### Step 1: Setup Development Environment

1. **Install Node.js and npm**: Ensure you have Node.js and npm installed. You can download and install them from [Node.js website](https://nodejs.org/).

2. **Install a Code Editor**: Use a code editor such as Visual Studio Code for better development experience.

3. **Create a New React/Next.js Project**: Use the command line to create a new project.

- For React: `npx create-react-app admin-dashboard`

- For Next.js: `npx create-next-app@latest admin-dashboard`

4. **Navigate to the Project Directory**: `cd admin-dashboard`

### Step 2: Create Initial Project Structure

1. **Setup Basic Pages**:

- `HomePage`

- `UsersListingPage`

- `PostsListingPage`

2. **Dummy Login Page**:

- Implement a simple login page with email and password inputs.

- On successful login (simulated with any credentials), navigate to the HomePage.

### Step 3: Implement Navigation

1. **Create a Navigation Bar**: Add a navigation bar on the left side of all pages.

- Links to HomePage, UsersListingPage, and PostsListingPage.

### Step 4: Build HomePage

1. **KPIs Display**: Show 4 KPIs in large rectangular boxes:

- Total Users

- Total Posts

- Users active in the last 24 hours

- Posts published in the last 24 hours

### Step 5: Build Users Listing Page

1. **KPIs**: Add two KPIs at the top.

- Total Users

- Users active in the last 24 hours

2. **Table View**: Display a paginated list of users.

- Columns: `User\_id`, `Username`, `Name`, `Email`

3. **Control Buttons**: Each row should have "Ban" and "Edit" buttons.

### Step 6: Build Posts Listing Page

1. **KPIs**: Display KPI information like on the Users Listing Page, if necessary.

2. **Table View**: Display a paginated list of posts.

- Columns: `post\_id`, `post caption`, `media url`

3. **Control Buttons**: Each row should have "Delete" and "Hide" buttons.

### Step 7: Create Dummy Data

1. **Users Data**: Simulate user data for the listing.

2. **Posts Data**: Simulate post data for the listing.

### Step 8: Add Styling

1. **CSS/Styled-components**: Add necessary styling to make the dashboard visually appealing.

2. **Responsive Design**: Ensure that the dashboard is responsive and operates well on different devices.

### Step 9: Test the Application

1. **Functional Testing**: Verify all features work as intended.

2. **Dummy Login**: Make sure any set of credentials can log in.

### Step 10: Documentation and Submission

1. **README File**: Write detailed steps for setting up and running the application.

2. **Screenshots**: Capture and include screenshots of the dashboard.

3. **Video Recording**: Create a 1-2 minute video demonstrating the dashboard.

4. **ZIP File**: Package all deliverables (code, README file, and video) into a ZIP file.

5. **Upload to Google Drive**: Upload the ZIP file to Google Drive and share the public access link.

## Best Practices

1. **Code Documentation**: Ensure your code is well-documented with comments explaining the logic.

2. **Component Reusability**: Create reusable components to avoid code duplication.

3. **Consistent Styling**: Use consistent CSS conventions or styled-components.

4. **Error Handling**: Implement error handling to manage unexpected scenarios gracefully.

## Submission Guidelines

1. **Ensure Code Completeness**: Make sure all required features are implemented and tested.

2. **README File**: Should include:

- Environment setup

- Dependencies

- Installation steps

- How to run the application

3. **Video Recording**: 1-2 minute video demonstrating the major functionalities.

4. **Screenshots**: Capture key sections of the dashboard.

5. **ZIP File**: Name the ZIP file appropriately (e.g., `your\_name\_admin\_dashboard.zip`), upload to Google Drive, and share the link.

## Frequently Asked Questions (FAQ)

**Q1: What if I encounter an issue with npm dependencies?**

- **A1**: Try deleting the `node\_modules` folder and `package-lock.json` file, then run `npm install` again. If the problem persists, check the specific error message and consult [Stack Overflow](https://stackoverflow.com).

**Q2: Can I use other libraries or frameworks?**

- **A2**: The assignment specifies Reactjs/Next.js. Stick to these frameworks to fulfill the requirements.

**Q3: How detailed should the styling be?**

- **A3**: While the primary focus is functionality, ensure the UI is clean, intuitive, and professional.

**Q4: How can I handle state management?**

- **A4**: Use React's built-in state management (useState, useEffect) or other libraries like Redux if needed.

**Q5: What if I'm unable to finish the assignment within the given time?**

- **A5**: It's important to communicate any delays as early as possible and provide a valid reason. Aim to submit a partially complete project that demonstrates your capabilities.

---

By following these instructions, you should be able to successfully complete the assignment while adhering to best practices and meeting all required deliverables.