FUNERAL DIRECTORS
H.PARSONS

Junior Developer Task: Funeral Service Venues Landing Page

Objective:

Demonstrate your ability to work with modern web development tools by creating a small web application to search and filter funeral service venues in the Illawarra (e.g. Churches & chapels, you can use dummy data to populate venues) The frontend should use React to fetch and display data. The backend should use SilverStripe CMS (PHP + MySQL database) to store and manage venue data.

Timeframe:

You have two weeks to complete the task from the date it is received.

Note:

You may choose to complete **Task 1** (frontend) or **Task 2** (backend). If you successfully complete both, proceed to **Task 3** to integrate the frontend and backend.

Task Overview

Task 1

1. **Build a React Frontend**:
    a. Create a basic, standalone React application. Utilise modern design themes & practices to create an aesthetically pleasing webpage – Material UI might be helpful.
    b. The application should include:
        i. A search bar and filters for venue name, capacity, or location.
        ii. Dynamic display of venue results in a clean, responsive layout. Each venue should show:
            1. Name
            2. Location
            3. Capacity

            4. Short description
        iii. Any other useful data, e.g. photos.
    c. The frontend should fetch venue data from a structured JSON file and use it to populate the search and filter functionality on-demand.

Task 2

2. **Set Up SilverStripe CMS**:
    a. Install and configure SilverStripe CMS version 4.
        i. Here is some documentation to help get you started with Silverstripe as well as lessons and several developer guides.
    b. Create custom DataObjects to represent funeral service venues. Each venue should have the following fields:
        i. Title (Text)
        ii. Address (Text)
        iii. Capacity (Integer)
        iv. Short Description (Text) – A brief overview of the venue.
        v. Venue Image(s)
    c. Use the CMS interface to manage all DataObjects.
    d. Display all the data on a simple web page. (No interactivity required.)

Task 3

3. **Integrate Frontend and Backend (Optional)**:
    a. If you complete both tasks above, connect the React frontend with the SilverStripe backend.
    b. Enable real-time data retrieval and filtering in the React frontend via GraphQL instead of reading from the static JSON file.

Deliverables

- A working standalone React application (if Task 1 is completed).
- A functional SilverStripe CMS with configured venues (if Task 2 is completed).
- A Git repository containing all code, with clear setup instructions in a README.md file.
- A demo presentation showcasing the features and explaining your approach – on your local machine is fine.

Supporting Material

Silverstripe:

1. See https://www.silverstripe.org/learn/lessons/v4/
    a. Lessons 0, 1 to get started
    b. Lessons 8, 9, 10 for structuring DataObjects
    c. Lesson 13 for managing data

React:

1. See https://react.dev/
    a. Getting started - https://react.dev/learn
    b. Fetching data - https://react.dev/learn/synchronizing-with-effects

Evaluation Criteria

- **Frontend Design & Functionality**: How well the React app is structured and performs. Intuitive design and user experience is also expected.
- **Backend Implementation**: Proper use of SilverStripe for managing and structuring data.
- **Integration**: If completed, how seamlessly the frontend communicates with the backend.
- **Code Quality**: Clean, modular, and well-documented code.
- **Problem-Solving**: Creativity and approach in completing the task.