**Documentation**

This documentation covers the header component, several other components as seen on the home page and a footer component.

**Steps to Construct the Page**  
The prepare of making the page, counting the improvement of components, layouts, models, controllers, and the integration with Sitecore, is portrayed in detail in this part.  
  
The page is built component by component and then all the components are stacked together in the desired orientation.   
Goal: To productively oversee the improvement handle, partition the page into its component parts.

**2. Start with the Three Main Components**

Components: Start with the page's main elements, which usually consist of the header and footer after two listing components.

Method: Determine the primary elements needed for the page.   
Since they serve as the foundation for the page structure, develop these elements first.

**3. Create Templates**

Create templates for each component to ensure a structured and organized approach. Develop templates for subitems within the listing components as well.

**Implementation:**

* Define the necessary fields for each template element.
* Ensure consistency across templates for related components

**4. Create Items Using Templates**

* Create items for each component using the predefined templates.
* For listing components, generate subitems to dynamically populate the lists.
* Access the Sitecore content tree through the Sitecore interface.
* Follow the structure defined in the templates to create items under the appropriate parent item.

**5. Develop Model, Controller Action Method, and View**

Create a model class for each component to represent the data structure.

**Controller:**

* Define a controller action method to fetch and process the required data.
* Ensure the controller action returns the model data to the view.

**View:**  
 Develop a view for each component to display the data on the frontend. Use Razor syntax to connect the model data to the view.

**6. Fetch Sitecore HTML Items and Implement Styling**

Utilize Sitecore's API to retrieve the required HTML elements. Bind these elements to the model within the controller action method.

**Styling:**

* Apply unique CSS styles stored in a designated folder.
* Ensure the styles are modular and reusable across multiple components.
* Use classes and IDs to target specific parts of the components effectively.

**7. Create Controller Rendering for the Component in Sitecore**

Set up a controller renderer for each Sitecore component. Define the controller, controller action, and corresponding view to render the component.

**8. Add Rendering of All Components to the Home Page in Presentation Details**

Access Sitecore and navigate to the Home page item. Edit the presentation details to add renderings for each component.

**Data Source and Placeholders:**

* Assign the appropriate data source to bind relevant content for each component.
* Specify the placeholders on the page where each component should be displayed.

**9. Make Necessary Components Cacheable**

Identify the components that can be cached to improve performance. Configure caching in Sitecore to store the output of these components.

**Implementation:**

* Set cacheable options such as checkboxes, cache duration, and vary-by parameters.
* Ensure that caching settings do not interfere with updates to dynamic content.

Below are the components in detail:

**Header Component**

**Description:** The header component consists of an external structure template with three primary items: a logo, a navigation bar (navbar), and a "Get Appointment" button.

**Templates:**

**Header component**

1. **Header External Structure Template:**
   * Logo
   * Home (Multilist)
   * Services (Multilist)
   * Blog (Multilist)
   * Pages (Multilist)
2. **Internal Structure Template:**
   * This is consisting of a link.

**It solution Component**

**Description:** The about us component required only one template as it did not involve a multilist. It included items such as image, preheader, title, description, and a button.

**Templates:**

1. **It solutions Template:**
   * Title
   * It solution text
   * Introduction
   * Getstarted
   * Video
   * BackgroundImage

**Service Component**

**Description:** The service component uses three items: a preheader, title, and a serviceCard item (a multilist component). Additionally, a separate template for serviceItems was created.

**Templates:**

1. **Service Component Template:**
   * Preheader
   * Title
   * Componentbox (Multilist)
   * Service button(General link)
2. **ServiceItem Template:**
   * Image
   * Title
   * Info
   * Link Text

**Testimonials Component**

**Description:** The testimonials component is a listing component with three items: a preheader, a title, and a multilist component for TestimonialCard. An internal template for the testimonial card was created with four items.

**Templates:**

1. **Testimonials Component Template:**
   * Preheader
   * Title
   * TestimonialItems (Multilist)
2. **TestimonialCard Template:**
   * Text
   * Image
   * Author and Designation

**Footer Component**

**Description:** The footer component is a listing component with an external structure template. It includes a multilist for four different columns, a droplink for the footer navbar, and a footer text.

**Templates:**

1. **Footer External Structure Template:**
   * Columns (Multilist)
   * Footer Text
2. **Single Column Template:**
   * Title(general link)
   * Link items (10 fields)