**REACT**

1. Create a layout
   1. Use the “UI Trait” (which is to specify a component name and css class names)
2. Create Page Template
   1. Use the “UI Trait”
   2. Optionally define which layout to use
3. Optionally create a route (if it is going to be a ‘static’ page)

**Front-End**

Page Templates and Layouts should look for a component group with a reference name. If it is not found, a message should appear with a button which will add the component group with the required reference using the API.

Layouts and page templates need the option to find and clear unused component groups.

When there is an empty component group, this should be clearly displayed to an admin user with the ability to add a component.

To add a component an admin will right click on desktop or long click on mobile for context menu. Context menu would have ‘add component’ option. This would send an API request to add a component location at the end and the component within it.

Reordering components (i.e. adjusting sort order of the component location) can be done with context menu, click reorder, re-orderable components shake like on mobiles for dragging and dropping. A done button will appear too.

For optional component groups (within components) this should appear as an option within the context menu. I.e. adding a component group to a navigation item which will then show the drop-down.

For a component, we need to know if it is a persisted component in the database, if not it cannot be edited or re-ordered as it is likely done by a data transformer.

Components which have editable areas should be named in the front-end and in context menu the option to edit ‘name’ will appear. I.e. the text/label of a navigation item. Once selected to edit, it’ll show as a text input and this can then be confirmed.

Components need option to ‘advanced edit’ – e.g. a navigation item needs to be able to set where it is linking to. This may need to be an internal route with an ID or external. The end user shouldn’t have to care! Perhaps auto-fill searching routes…

Each component as it is added/modified by the front-end should be made a draft. Validation for components in draft state should be carried out but not enforced – ‘not null’ to be discussed if database enforcing validation. Component in draft state will have a badge, context menu will allow to publish changes or set a date/time for changes to publish. The API will need to detect whether there is a draft component linked to each component on output, if so, check the published date/time if set and proceed to publish the draft component to the main component and delete the draft.

There will still be a main admin bar still to publish all changes and edit the page data similar to now.

As an **admin**, any page template should be able to be loaded using special routing. E.g. [www.websites.com/\_page/\_\_ID\_\_](http://www.websites.com/_page/__ID__) This is because we may add a page that uses a data transformer to put the data into the page. E.g. a news article. When a dynamic page is outputted, we need to map back to the persisted data entity to modify that content and not the component directly…

For news articles / dynamic data – it is the entire page that is a draft/published. This is because the database entity is the news article.

A component could list pre-set class names that will adjust the styling a little bit in the ‘advanced edit’.

Component groups need to have ‘allowed components’ e.g. no hero in the navigation.

Components may restrict which component groups they are allowed in???? Maybe specify if component needs to be explicitly allowed in a component group.