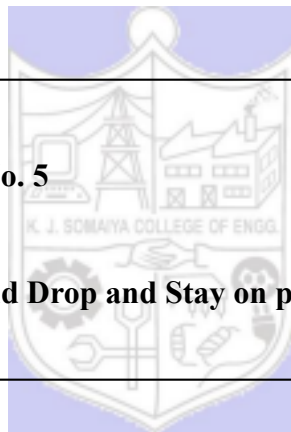


Experiment No. 5

Title: Drag and Drop and Stay on page principles



Aim: To create wireframe for Web UI –Drag and Drop and stay on page principles

Resources needed: Wireframing tool

Theory:

Drag and Drop

There are at least 15 events available for cueing the user during a drag and drop interaction:

Page Load: Before any interaction occurs, you can pre-signify the availability of drag and drop. For example, you could display a tip on the page to indicate draggability.

Mouse Hover: The mouse pointer hovers over an object that is draggable.

Mouse Down: The user holds down the mouse button on the draggable object.

Drag Initiated: After the mouse drag starts (usually some threshold—3 pixels).

Drag Leaves Original Location: After the drag object is pulled from its location or object that contains it.

Drag Re-Enters Original Location: When the object re-enters the original location.

Drag Enters Valid Target: Dragging over a valid drop target.

Drag Exits Valid Target: Dragging back out of a valid drop target.

Drag Enters Specific Invalid Target: Dragging over an invalid drop target.

Drag Is Over No Specific Target: Dragging over neither a valid or invalid target. Do you treat all areas outside of valid targets as invalid?

Drag Hovers Over Valid Target: User pauses over the valid target without dropping the object. This is usually when a spring loaded drop target can open up. For example, drag over a folder and pause, the folder opens revealing a new area to drag into.

Drag Hovers Over Invalid Target: User pauses over an invalid target without dropping the object.

Drop Accepted: Drop occurs over a valid target and drop has been accepted.

Drop Rejected: Drop occurs over an invalid target and drop has been rejected. Do you zoom back the dropped object?

Drop on Parent Container: Is the place where the object was dragged from special? Usually this is not the case, but it may carry special meaning in some contexts.

During each event one can visually manipulate a number of actors. The page elements available include:

- Page (e.g., static messaging on the page)
- Cursor
- Tool Tip
- Drag Object (or some portion of the drag object, e.g., title area of a module)
- Drag Object's Parent Container
- Drop Target

Stay on page Principle includes:**• Overlays**

Instead of going to a new page, a mini-page can be displayed in a lightweight layer over the page.

• Inlays

Instead of going to a new page, information or actions can be inlaid within the page.

• Virtual Pages

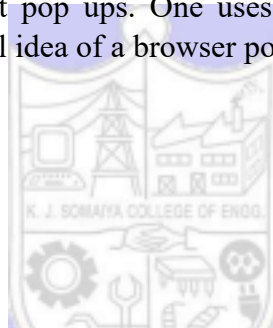
By revealing dynamic content and using animation, we can extend the virtual space of the page.


Overlay

Overlays are really just lightweight pop ups. One uses the term lightweight to make a clear distinction between it and the normal idea of a browser pop up.

The three specific types of overlays:


- Dialog Overlays,
- Detail Overlays, and
- Input Overlays

Dialog Overlay



Activation

Clicking the "Buy" button initiates the purchase process.




Overlay treatment

The confirmation dialog is shown in a lightweight overlay. Since the overlay is modal (interaction is only accepted in the overlay) the rest of the page is dimmed down. The user may also cancel the purchase.

Figure Netflix uses a lightweight pop up to confirm a previously viewed DVD purchase; in addition, it uses the Lightbox Effect to indicate modality

Detail Overlay

The Detail Overlay allows an overlay to present additional information when the user clicks or hovers over a link or section of content. Toolkits now make it easier to create overlays across different browsers and to request additional information from the server without refreshing the page.



Detail overlay activation

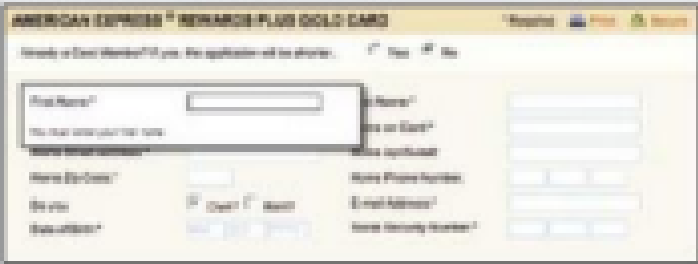
However, often more information is needed to decide whether a movie should be played or added to a movie queue.

By providing a synopsis along with personalized recommendation information, the user can quickly make a determination.

The movie detail information is displayed after a slight delay.

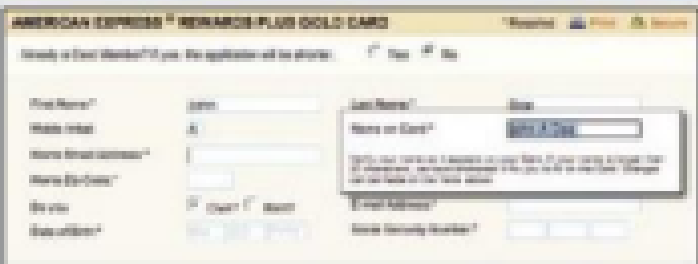
Input Overlay

Input Overlay is a lightweight overlay that brings additional input information for each field tabbed into.




Input overlay

Tabbing or clicking into any field wraps the field in an overlay. The overlay provides additional input information.



Obscuring fields

The overlay does obscure fields just below it, but not to the left or right.



Deactivation

Clicking anywhere removes the overlay. This lets the user click through the field covered by the overlay.

Inlays

A simple technique is to expand a part of the page, revealing a dialog area within the page.



List Inlays

Lists are a great place to use Inlays. The List Inlay works as an effective way to hide detail until needed.

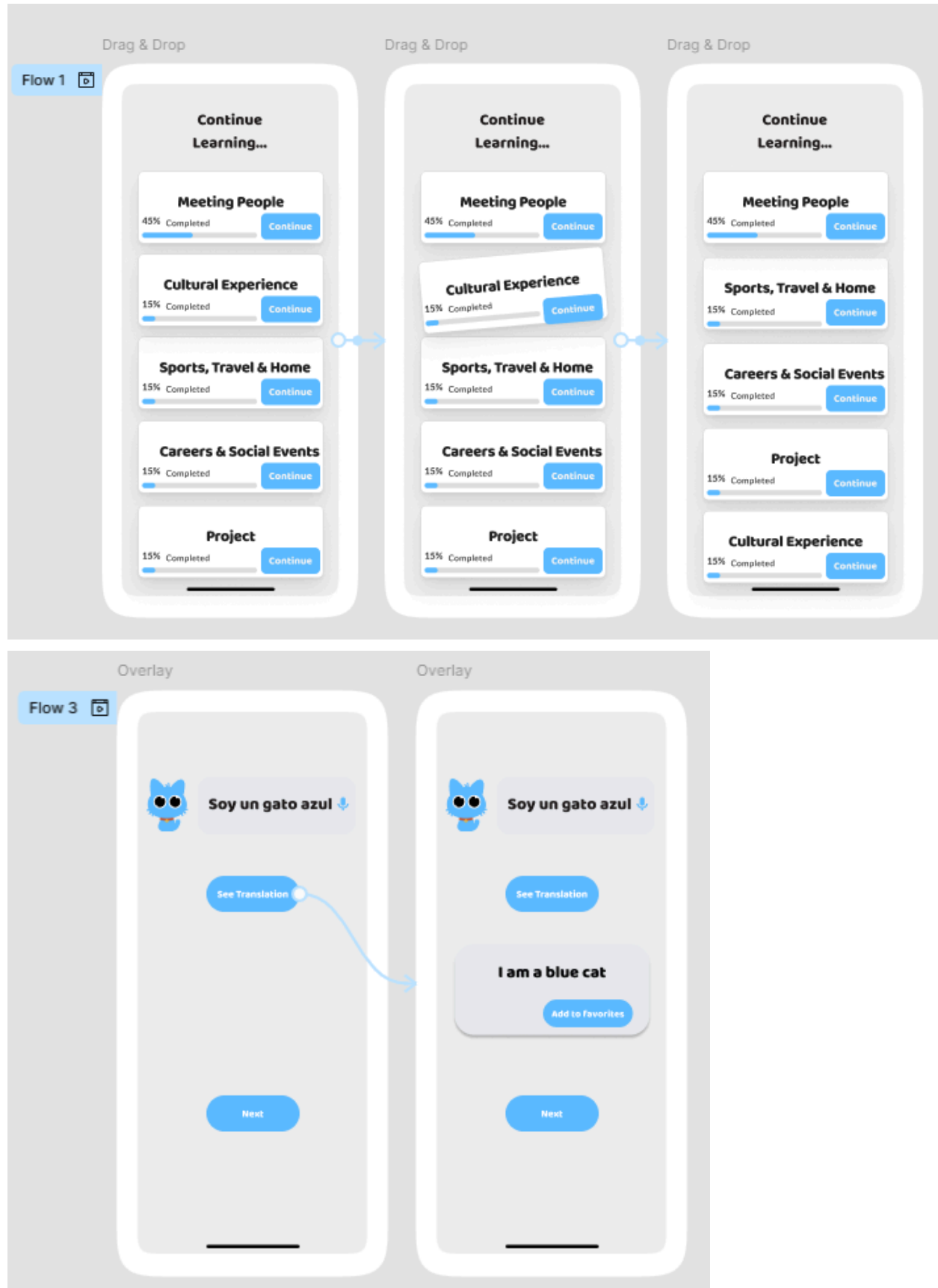


Virtual Page

- Overlays allow bringing additional interactions or content in a layer above the current page. Inlays allow doing this within the page itself.
- However, another powerful approach to keeping users engaged on the current page is to create a virtual page.
- Patterns that support virtual pages include:
 - Virtual Scrolling
 - Inline Paging
 - Scrolled Paging Carousel
 - Panning
 - Zoomable User Interface

Procedure:

- Create wireframes incorporating Drag and Drop and Stay on page principles for chosen topic

Result:

Question:**1. Discuss in detail any one web development framework/technology to implement drag and drop principle.**

Answer: One of the most widely used frameworks for implementing the drag and drop principle in web development is React DnD (React Drag and Drop). React DnD is a set of React components and utilities that enable developers to easily implement drag-and-drop functionality in React applications. React DnD uses the HTML5 drag-and-drop API, abstracting its complexities to provide a simpler interface. This library allows for a highly flexible and customizable drag-and-drop experience.

Here's how React DnD works:

1. **Drag Sources:** A component that can be dragged is defined as a "drag source." You can assign different types to drag sources and control what happens when they are dragged.
2. **Drop Targets:** A drop target is an area where a draggable item can be dropped. You define the actions that should occur when an item is dropped onto the target, such as updating the application state or providing visual feedback.
3. **Backend:** React DnD uses an abstraction called a "backend." The default backend is based on the HTML5 drag-and-drop API, but there are other backends for different environments, such as touch devices or different browser settings.

Key Benefits of React DnD:

- **Customizability:** React DnD offers complete flexibility over what should happen during each drag and drop event.
- **Decoupled Logic:** Drag-and-drop logic is decoupled from the component rendering, making the code cleaner and more maintainable.
- **Integration with Redux:** React DnD integrates well with state management libraries like Redux, which makes it easier to manage complex drag-and-drop interactions that affect the global state of the application.

Implementation Example: To implement a simple drag-and-drop feature, you would wrap components with `DragSource` and `DropTarget` from the React DnD library, providing configurations and callback functions for handling the drag-and-drop events.

```
npm install react-dnd react-dnd-html5-backend
```

After installation, you can use React DnD components and hooks to implement a draggable interface.

```
import { useDrag, useDrop } from 'react-dnd';

function DraggableItem({ id, name }) {
  const [{ isDragging }, drag] = useDrag({
    type: 'ITEM',
    item: { id },
    collect: (monitor) => ({
```

```

        isDragging: !!monitor.isDragging(),
      )),
    ));

    return (
      <div ref={drag} style={{ opacity: isDragging ? 0.5 : 1 }}>
        {name}
      </div>
    );
  }

function DropTarget({ onDrop }) {
  const [, drop] = useDrop({
    accept: 'ITEM',
    drop: (item) => onDrop(item.id),
  });

  return <div ref={drop} style={{ height: '200px', border: '1px solid
black' }}>Drop here</div>;
}

```

This simple example creates a draggable item and a drop target where the item can be dropped.

Outcomes: Apply principles of Web interface design

Conclusion: (Conclusion to be based on the objectives and outcomes achieved)

The wireframe and design process using drag-and-drop and stay-on-page principles have been successfully demonstrated. The wireframe includes the ability to drag and drop objects while providing feedback during each step of the interaction. Additionally, the "stay on page" principle is implemented using overlays, inlays, and virtual pages to ensure the user remains engaged on the same page without unnecessary navigation, enhancing usability and user experience. The objective of applying web interface design principles was achieved by understanding the various events in the drag-and-drop interaction and employing techniques to keep users focused on the current page.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

(A Constituent College of Somaiya Vidyavihar University)

References:

1. Wilbert O. Galitz, “The Essential Guide to User Interface Design - An Introduction to GUI Design Principles and Techniques”, Wiley Computer Publishing, Second Edition, 2002
 2. Bill Scott, Theresa Neil, “Designing Web Interfaces Principles & Patterns for Rich Interaction”, O’rielly Media, First Edition, 2009
-