Epic 2.3: Layout Components

Epic Overview

This epic establishes the foundational layout components for THE WHEEL design system, providing responsive layout primitives and structural components that work across all workspace contexts.

Priority: P0 (Critical)
Timeline: 2 weeks
Dependencies: Epic 1.1 (Monorepo Architecture Setup), Epic 2.1 (Input Components), Epic 2.2 (Display Components)

Story 2.3.1: Layout Primitives

Overview

Enhance layout primitive components (Container, Grid, Flex, Stack) for responsive design and workspace context support.

AI Developer Prompt

You are enhancing layout primitive components (Container, Grid, Flex, Stack) for THE WHEEL design system. Building on the media and visual components from Story 2.2.3, you need to create responsive layout primitives that work across workspace contexts.

Context

- Enhanced media and visual components with workspace context
- Existing Container, Grid, Flex, Stack components
- Need responsive layout support for all device sizes
- Must integrate with workspace context theming
- Layout primitives are foundation for all other components

Requirements

1. Enhance Container component with workspace variants:

- Workspace context styling
- Responsive breakpoint handling
- Max-width variants for different contexts

- Padding and margin workspace theming
- Fluid and fixed width options

2. Enhance Grid component with responsive system:

- CSS Grid implementation with fallbacks
- Responsive grid columns and gaps
- Workspace context spacing
- Auto-fit and auto-fill options
- Grid template areas support

3. Enhance Flex component with workspace spacing:

- Flexbox utilities with workspace theming
- Responsive flex properties
- Gap support with workspace spacing
- Alignment utilities
- Wrap and direction controls

4. Enhance Stack component variants:

- Vertical and horizontal stacking
- Workspace context spacing
- Responsive stack behavior
- Separator integration
- Alignment options

Specific Tasks

- [ ] Enhance Container with workspace variants
- [ ] Implement responsive grid system
- [ ] Add workspace spacing to Flex component
- [ ] Create Stack component variants
- [ ] Implement responsive utilities
- [ ] Add workspace context spacing API
- [ ] Create layout documentation

Documentation Required

- Layout primitive system architecture
- Responsive layout implementation guide
- Workspace context spacing system
- Grid and flexbox usage guidelines
- Layout composition patterns
- Accessibility implementation details

Testing Requirements

- Responsive layout behavior tests
- Workspace context spacing tests
- Grid system functionality tests
- Flexbox behavior tests
- Stack component tests
- Cross-browser compatibility tests
- Performance tests for layout rendering

Integration Points

- Integration with workspace context providers
- Responsive design system integration
- Spacing system integration
- Theme system CSS variable integration
- Container query integration

Deliverables

- Enhanced Container component with workspace variants
- Responsive Grid system implementation
- Flex component with workspace spacing
- Stack component with alignment options
- Responsive layout utilities
- Comprehensive Storybook stories

Component Specifications

typescript

```tsx
interface ContainerProps extends React.HTMLAttributes<HTMLDivElement> {
  size?: 'xs' | 'sm' | 'md' | 'lg' | 'xl' | '2xl' | 'full'
  context?: 'consultant' | 'client' | 'admin' | 'neutral'
  responsive?: boolean
  padding?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  margin?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  fluid?: boolean
  centerContent?: boolean
  children: React.ReactNode
}

interface GridProps extends React.HTMLAttributes<HTMLDivElement> {
  columns?: number | string
  rows?: number | string
  gap?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  context?: 'consultant' | 'client' | 'admin' | 'neutral'
  responsive?: boolean
  autoFit?: boolean
  autoFill?: boolean
  templateAreas?: string
  children: React.ReactNode
}

interface FlexProps extends React.HTMLAttributes<HTMLDivElement> {
  direction?: 'row' | 'column' | 'row-reverse' | 'column-reverse'
  wrap?: 'nowrap' | 'wrap' | 'wrap-reverse'
  justify?: 'start' | 'end' | 'center' | 'between' | 'around' | 'evenly'
  align?: 'start' | 'end' | 'center' | 'stretch' | 'baseline'
  gap?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  context?: 'consultant' | 'client' | 'admin' | 'neutral'
  responsive?: boolean
  children: React.ReactNode
}

interface StackProps extends React.HTMLAttributes<HTMLDivElement> {
  direction?: 'vertical' | 'horizontal'
  spacing?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  context?: 'consultant' | 'client' | 'admin' | 'neutral'
  align?: 'start' | 'end' | 'center' | 'stretch'
  separator?: React.ReactNode
  responsive?: boolean
```

```
  children: React.ReactNode
}
```

Story 2.3.2: Structural Components

Overview

Enhance structural components (Card, Panel, Separator) for sophisticated content containers across workspace contexts.

AI Developer Prompt

You are enhancing structural components (Card, Panel, Separator) for THE WHEEL design system. Building on the layout primitives from Story 2.3.1, you need to create sophisticated structural components that provide consistent content containers across workspace contexts.

Context

- Enhanced layout primitives with workspace context
- Existing Card, Panel, Separator components
- Need sophisticated content containers for workspace applications
- Must support interactive states and animations
- Structural components are used throughout the application

Requirements

1. Enhance Card component with workspace variants:

- Workspace context styling and theming
- Interactive states (hover, focus, active)
- Elevation system with workspace shadows
- Card composition with header, body, footer
- Action integration and click handling

2. Enhance Panel component with workspace context:

- Collapsible panel functionality
- Workspace context styling
- Panel header with actions
- Resizable panels for layouts

- Responsive panel behavior

3. Enhance Separator component:

- Workspace context styling
- Vertical and horizontal orientations
- Text and icon separator variants
- Responsive separator behavior
- Integration with spacing system

Specific Tasks

- [ ] Enhance Card with workspace variants
- [ ] Add interactive states to Card
- [ ] Implement Panel with workspace context
- [ ] Add collapsible functionality to Panel
- [ ] Enhance Separator with workspace styling
- [ ] Create elevation system for Cards
- [ ] Implement responsive behavior

Documentation Required

- Structural component system architecture
- Card composition patterns and usage
- Panel functionality and responsive behavior
- Separator usage guidelines
- Interactive state implementation
- Accessibility implementation details

Testing Requirements

- Card interactive state tests
- Panel collapsible functionality tests
- Separator display tests
- Workspace context styling tests
- Responsive behavior tests
- Accessibility compliance tests
- Performance tests for animations

Integration Points

- Integration with workspace context providers
- Animation system integration
- Elevation system integration
- Theme system CSS variable integration
- Interactive state management integration

Deliverables

- Enhanced Card component with workspace variants
- Panel component with workspace context
- Separator component with workspace styling
- Interactive state system
- Elevation system implementation
- Comprehensive Storybook stories

Component Specifications

typescript

```typescript
interface CardProps extends React.HTMLAttributes<HTMLDivElement> {
  variant?: 'elevated' | 'outlined' | 'filled'
  context?: 'consultant' | 'client' | 'admin' | 'neutral'
  elevation?: 0 | 1 | 2 | 3 | 4 | 5
  interactive?: boolean
  padding?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  radius?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  header?: React.ReactNode
  footer?: React.ReactNode
  onClick?: () => void
  onHover?: () => void
  children: React.ReactNode
}

interface PanelProps extends React.HTMLAttributes<HTMLDivElement> {
  variant?: 'elevated' | 'outlined' | 'filled'
  context?: 'consultant' | 'client' | 'admin' | 'neutral'
  collapsible?: boolean
  defaultCollapsed?: boolean
  header?: React.ReactNode
  actions?: React.ReactNode
  resizable?: boolean
  onCollapse?: (collapsed: boolean) => void
  onResize?: (size: number) => void
  children: React.ReactNode
}

interface SeparatorProps extends React.HTMLAttributes<HTMLDivElement> {
  orientation?: 'horizontal' | 'vertical'
  context?: 'consultant' | 'client' | 'admin' | 'neutral'
  spacing?: 'none' | 'sm' | 'md' | 'lg' | 'xl'
  variant?: 'solid' | 'dashed' | 'dotted'
  text?: string
  icon?: React.ReactNode
  decorative?: boolean
}
```

Timeline and Dependencies

Timeline

- Week 1: Story 2.3.1 - Layout Primitives
- Week 2: Story 2.3.2 - Structural Components

Dependencies

- Epic 1.1 (Monorepo Architecture Setup) - Complete
- Epic 2.1 (Input Components) - Should be complete
- Epic 2.2 (Display Components) - Should be complete
- Workspace context system operational
- Spacing system defined

Success Metrics

- All layout components support responsive design
- Workspace context spacing implemented
- Performance benchmarks met (rendering under 16ms)
- 100% accessibility compliance
- Complete test coverage (90%+ for all components)
- Consistent layout behavior across browsers

Risk Mitigation

- Test on multiple devices and screen sizes
- Performance monitoring for layout rendering
- Browser compatibility testing
- Clear documentation for layout patterns
- Migration guides for existing layouts
- Regular design reviews for consistency