Example Workflow

Input SVG (Before):

```
<svg width="300" height="300" xmlns="http://www.w3.org/2000/svg">
<rect id="hero" x="50" y="50" width="200" height="100" fill="red"/>
<circle cx="150" cy="200" r="40" fill="green"/>
<rect x="20" y="20" width="50" height="50" fill="blue" class="small-box"/>
</svg>
```

User Instruction: Apply a diagonal gradient from #123456 to #abcdef to all circles and give the element with id 'hero' a radial gradient from red to white

Parsed Gradient Config:


```
{
 "steps": [
  {
   "targets": [{"selector": "circle", "description": "all circles"}],
   "gradient": {
    "type": "linear",
    "direction": "diagonal",
    "stops": [
     {"offset": 0, "color": "#123456"},
     {"offset": 100, "color": "#abcdef"}
    ]
   }
  },
  {
```

```
    "targets": [{"selector": "#hero", "description": "element with id 'hero'"}],
    "gradient": {
      "type": "radial",
      "direction": "horizontal",
      "stops": [
        {"offset": 0, "color": "#ff0000"},
        {"offset": 100, "color": "#ffffff"}
      ]
    }
  }
 ]
}
```


Logs (Agent Workflow):

1. Instruction Analysis Phase: ✅ Parsed configuration with 2 steps

2. SVG Modification Phase: 🎨 Created gradients and applied to target elements

3. Quality Validation Phase: ✅ Validation successful!

Output SVG (After):


```
<svg width="300" height="300" xmlns="http://www.w3.org/2000/svg">
 <defs>
  <linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="100%">
   <stop offset="0%" style="stop-color:#123456;stop-opacity:1" />
   <stop offset="100%" style="stop-color:#abcdef;stop-opacity:1" />
  </linearGradient>
  <radialGradient id="grad2" cx="50%" cy="50%" r="50%">
   <stop offset="0%" style="stop-color:#ff0000;stop-opacity:1" />
```

```
    <stop offset="100%" style="stop-color:#ffffff;stop-opacity:1" />
   </radialGradient>
  </defs>
  <rect id="hero" x="50" y="50" width="200" height="100" fill="url(#grad2)"/>
  <circle cx="150" cy="200" r="40" fill="url(#grad1)"/>
  <rect x="20" y="20" width="50" height="50" fill="blue" class="small-box"/>
</svg>
```

Agents & Their Logic:

- **Gradient Parser Agent**
  - Interprets user instructions (LLM or regex fallback).
  - Extracts gradient type, direction, colors, and targets.
- **SVG Modifier Agent**
  - Generates <linearGradient> / <radialGradient> definitions.
  - Applies them to matching elements (by id, class, or tag).
- **Integrity Checker Agent**
  - Ensures all gradient references are defined.
  - Validates overall SVG structure.

Goal:

To demonstrate how CrewAI agents can handle design automation by transforming natural language into structured, validated SVG updates.