Future Program Support Plan

1. **localStorage Based Persistence**
   a. Implementation:
      i. Create a function to save the current state of the shapes to the browser's localStorage.
      ii. Create a function to load the shapes from localStorage when the application starts.
      iii. When adding, updating, or removing shapes, update the localStorage accordingly.
   b. How Current Implementation Helps:
      i. The current implementation, which maintains a state for shapes, makes it relatively easy to implement localStorage-based persistence.
      ii. We can serialize the shapes state into JSON and save it in localStorage. Loading can be done by deserializing the stored JSON.
2. **Undo/Redo Functionality**
   a. Implementation:
      i. Maintain a history of the shapes state after each action (change of location/properties).
      ii. Create functions for undo and redo actions that revert or reapply changes to the shapes state based on the history.
   b. How Current Implementation Helps:
      i. The current implementation, which already manages state changes through functions like $addShape$, $onUpdateShape$, and $setShapes$, can be extended to maintain a history of these changes.
      ii. We can use the existing shapes state to build a history of changes and apply them during undo/redo operations.
3. **Save to Image**
   a. Implementation:
      i. Create a function that generates an image from the canvas content.
      ii. Allow users to choose an image format (e.g., PNG or JPEG).
      iii. Provide an option to download the generated image.
   b. How Current Implementation Helps:
      i. The current implementation uses a $<canvas>$ element to draw shapes, making it suitable for exporting as an image.
      ii. The canvas content can be converted into an image in various formats (e.g., using the $toDataURL$ method).
4. **Additional Features**
   a. The current implementation's modular structure, with separate components for the canvas, shape list, and property editor, makes it easier to add new features without extensive code changes.