SOFTWARE ARCHITECTURE DESIGN
GRUPPO 10

Report of the Group 10, whose members are:
Izzo Mario, Maruotto Alessandro, Nocerino Pierluigi Pio, Paolillo Sabato

Trello Board:
https://trello.com/invite/b/681b8667cc462d5e80db591c/ATTI253c3d1470f8c568e85448581b5056040B7D4DA9/gruppo10-sad

GitHub Repository:
https://github.com/m-izzo72/SoftwareArchitectureDesign

PREGAME

USER EPIC

1. **Basic operation of the program**
   - The program is used to make drawings composed of one or more geometrical shapes. The user is presented with a window that is initially empty.
   - The user can choose a geometrical shape to add; using the mouse on the window, the shape is added on the selected position.
   - The shapes supported must include line segments, rectangles, ellipses.
   - The user must be able to choose the colour of the shape to be added (at least 8 different colours must be supported). Also, if the shape is a closed contour (e.g. a rectangle), the user must be able to choose the colour of the interior of the shape.
   - The user must be able to save the drawing on a file, and to load a drawing previously saved.

2. **Shape editing**
   - The user can select a shape of the drawing with a mouse operation. The user can delete the selected shape, can move it to a different position, can change its colour(s) or its size.
   - The user can perform the Cut, Copy and Paste operations (as normally found in programs) with shapes.
   - All the operations that change the drawing must be undoable, with an unlimited number of undo levels.
   - The user can send a shape "to the front" or "to the back" of the drawing; this has a visible effect when two or more shapes are overlapping.

3. **Display of the drawing**
   - The user must be able to change the zoom level with which the drawing is shown in the user interface. At least 4 different zoom levels must be supported.
   - The user interface must support a drawing surface that is larger than the size of the window; the user can scroll to the portion of the drawing of interest.
   - The user interface can display an optional grid to help the user positioning the shapes. The user must be able to turn the grid on or off, and also to choose the size of the grid.

4. **More shapes and editing operations**
   - The user must be able to enter an arbitrary polygon (not just a regular one) as a shape. The user must be able to enter a text string as a shape (in this case, also the user can choose the size of the characters).
   - The user must be able to rotate a shape by an arbitrary angle (i.e. not necessarily a multiple of 90°). The user must be able to "mirror" a shape horizontally or vertically. The user must be able to "stretch" a shape, horizontally or vertically.

5. **Groups and shape libraries**
   - The user must be able to select more than one shape, and then "group" the selected shapes. The created group will behave as a single shape in successive operations (e.g. when selected moved). The user must be able to "ungroup" a previously grouped set of shapes.
   - The user must be able to add new "shape creation commands" to the application: the user selects a shape and gives it a name; later, when the user executes the corresponding shape creation command, the program adds to the drawing a new copy of that shape (note that the shape might have been removed from the drawing, or might have been modified; the command must create a copy that reproduces the shape as it was when the user created the new command).
   - The "shape creation commands" must be saved within the drawing file. Additionally, the user must be able to "export" the shape creation commands to a "shapes library file", or to "import" them from such a file (for instance, the user may want to import a library of shapes representing electrical components).

PRODUCT BACKLOG

ID	TITLE	DESCRIPTION	ACCEPTANCE CRITERIA	PRIORITY	SP
1.1	Empty window on start-up	As a user, I want an empty window when I open the application, So that I can start drawing from scratch	**AC1**:Given the application is properly installed, When I launch it, Then the application should start without any runtime errors or warnings. **AC2:** Given the application is started, When it loads, Then a window should be visible to the user. **AC3**: Given the application starts, When the window appears, Then the drawing area should be completely empty (no shapes, text, or UI artifacts). **AC4**: Given the application window is open, Then the title bar should display “New Canvas”.	High	1
1.2	Line drawing	As a user,	**AC1**: Given that I have selected the "line" tool,	High	2

		I want to be able to draw a line segment, So that I can create drawings composed of this shape	When I click on a first point in the drawing area, Then the starting point of the line is recorded. **AC2**: Given that a starting point has been selected, When I drag to a second point in the drawing area, Then a straight line segment is drawn connecting the two points. **AC3**: Given that the line has been drawn, Then it becomes part of the drawing and remains visible.		
1.3	Rectangle drawing	As a user, I want to be able to draw a rectangle, So that I can create drawings composed of this shape	**AC1**: Given that I have selected the "rectangle" tool, When I click on a point in the drawing area, Then that point is recorded as one corner of the rectangle. **AC2**: Given that a first corner has been selected, When I drag to a second point in the drawing area, Then a rectangle is drawn using the two points as opposite corners. **AC3**: Given that the rectangle has been drawn, Then it becomes part of the drawing and remains visible.	High	2
1.4	Ellipse drawing	As a user, I want to be able to draw an ellipse, So that I can create drawings composed of this shape	**AC1**: Given that I have selected the "ellipse" tool, When I click a first point in the drawing area, Then it is recorded as one of the corner the ellipse is inscribed into. **AC2**: Given that the first point is set, When I drag to a second point, Then it is recorded as the second corner of the rectangle the ellipse is inscribed into. **AC3**: Given that both points have been set, When I release the mouse button, Then the ellipse is drawn such that it is inscribed in the rectangle made with the previous points. **AC4**: Given that the ellipse is drawn, Then it becomes part of the drawing and remains visible.	High	2
1.5	Shape colour	As a user, I want to choose a colour for the shape (from the	**AC1**: Given that I have selected a colour from the available options	Medium-High	2

		following options: black, white, red, green, blue, yellow, teal, purple), So that I can create drawings composed of coloured shapes	(black, white, red, green, blue, yellow, teal, purple), When I draw a shape (line, rectangle, or ellipse), Then the shape is rendered in the selected colour. **AC2**:Given that no colour has been selected, When I draw a shape, Then a default colour (e.g., light blue) is used. **AC3**: Given that I have selected a new colour, When I draw additional shapes, Then each new shape uses the newly selected colour.		
1.6	Border colour	As a user, I want to choose a border colour for the shape (from the following options: black, white, red, green, blue, yellow, teal, purple), So that I can create drawings composed of coloured shapes	**AC1**:Given that I have selected a border colour from the available options (black, white, red, green, blue, yellow, teal, purple), When I draw a shape (line, rectangle, or ellipse), Then the border of the shape is rendered in the selected colour. **AC2**: Given that no border colour has been explicitly selected, When I draw a shape, Then a default border colour (e.g., blue) is applied. **AC3**: Given that I change the border colour selection, When I draw a new shape, Then the new shape's border uses the newly selected colour.	Medium-High	2
1.7	Saving the drawing in a file	As a user, I want to save a drawing into a file, So that I can access it whenever I want	**AC1**: Given that I have created a drawing*, When I click the "Save" button, Then the current drawing is saved as a file on my device. **AC2**: Given that I save the drawing, Then the file should have a default name (e.g., canvas).	High	8
1.8	Opening a drawing via file	As a user, I want to load a drawing from a file, So that I can keep drawing on a previously saved canvas	**AC1**: Given that I have a valid drawing file saved on my device, When I click the "Open file" button and select the file, Then the drawing is loaded and displayed on the canvas. **AC2**: Given that I load a drawing, Then all the shapes (lines, rectangles, ellipses, colours, borders, etc.) appear as they were at the time of saving.	High	8

			AC3: Given that I open a file of unsupported format or invalid content, Then the application shows an appropriate error message and does not crash.		
2.1	Shape of the drawing	As a user, I want to select a shape of the drawing with a mouse operation, So that I can freely edit it	**AC1**: Given that I have created one or more shapes on the canvas, When I click on a shape with the mouse, Then that shape becomes selected. **AC2**: Given that I click outside any shape, Then any currently selected shape becomes deselected.	High	1
2.2	Delete operation	As a user, I want to be able to delete a shape of the drawing, So that I can remove an element I no longer need	**AC1**: Given that I have created one or more shapes on the canvas and I have selected a shape, When I click the "Delete" button, Then the selected shape is removed from the canvas. **AC2**: Given that no shape is selected, When I click the "Delete" button, Then nothing happens. **AC3**: Given that I delete a shape, Then the shape is permanently removed and no longer visible or editable.	High	2
2.3	Shape preview	As a user, I want to see a shape preview I'm drawing. So that I can place it precisely before finalizing it	**AC1**: Given that I have selected a shape tool (e.g., line, rectangle, ellipse), When I click and start dragging the cursor on the canvas, Then a preview of the shape appears and updates in real-time as I move the cursor. **AC2**: Given that a shape preview is active, Then it is rendered with a distinct visual style (e.g., dashed border, semi-transparent fill) to indicate that it is not final. **AC3**: Given that I release the mouse button (or complete the drawing action), Then the preview becomes a finalized shape, rendered normally on the canvas.	High	3
2.4	Move operation	As a user, I want to be able to move a shape of the drawing, So that I can freely adjust my drawing layout	**AC1**: Given that I have created one or more shapes on the canvas and I have selected a shape, When I click and drag the shape with the mouse,	High	5

			Then the shape follows the cursor and moves accordingly. **AC2**: Given that I am dragging a shape, When I release the mouse button, Then the shape is placed at its final position on the canvas. **AC3**: Given that no shape is selected, When I click and drag on the canvas, Then no shape is moved. **AC4**: Given that I move a shape, Then its visual style (e.g., colour, border) remains unchanged. **AC5**: Given that I move a shape near the edge of the canvas, Then the shape cannot be dragged outside the visible drawing area		
2.5	Changing colour operation	As a user, I want to be able to change the colour of a shape (or its border) of the drawing, So that I can further customize its appearance	**AC1**:Given that I have created a shape and selected it, When I click the "Change fill colour" button and choose a new colour, Then the fill of the selected shape is updated with the new colour. **AC2**:Given that I have selected a shape, When I click the "Change border colour" button and choose a new colour, Then the border of the selected shape is updated with the new colour. **AC3**: Given that no shape is selected, When I attempt to change a colour, Then no change is applied and the application may show a message like "No shape selected". **AC4**: Given that I have selected a shape with an existing fill and/or border colour, Then changing the colour updates only the selected property (fill or border), not both unless explicitly done. **AC5**:Given that I change the colour of a shape, Then the shape remains in the same position and retains its other properties	Medium-High	2

2.6	Changing size operation	As a user, I want to be able to change the size of a shape of the drawing, So that I can further customize its appearance	**AC1**: Given that I have created a shape and selected it, When I click and drag one of its side or top/bottom or lateral handles, Then the shape is resized freely according to the direction of the drag. **AC2**: Given that I have selected a shape, When I click and drag from one of its corner handles, Then the shape is resized while maintaining its original proportions (aspect ratio is locked). **AC3**: Given that a shape is selected, Then resize handles are visible and enabled around the shape. **AC4**: Given that I resize a shape, Then all other properties of the shape (e.g., fill colour, border, type) remain unchanged. **AC5**: Given that I resize a shape, Then the resizing should be visible in real time as I drag. **AC6**: Given that no shape is selected, When I attempt to resize, Then no resizing occurs.	Medium-High	8
2.7	Cut operation	As a user, I want to be able to cut a shape of the drawing, So that I can paste it elsewhere whenever I want	**AC1**: Given that I have created a shape and selected it, When I click the "Cut" button, Then the selected shape is removed from the canvas and stored in the clipboard. **AC2**: Given that a shape has been cut, Then it can be pasted elsewhere using the "Paste" operation. **AC3**: Given that no shape is selected, When I click the "Cut" button, Then nothing happens and no content is added to the clipboard. **AC4**: Given that I cut a shape, Then all its properties (e.g., size, position, fill colour, border) are preserved in the clipboard for later use.	High	2
2.8	Copy operation	As a user, I want to be able to copy a shape of the drawing,	**AC1**: Given that I have created a shape and selected it, When I click the "Copy" button,	High	2

		So that I can paste it elsewhere whenever I want	Then a copy of the selected shape is added to the clipboard without removing it from the canvas. **AC2**: Given that a shape has been copied, Then I can paste it onto the canvas using the "Paste" operation. **AC3**: Given that no shape is selected, When I click the "Copy" button, Then nothing happens and the clipboard remains unchanged. **AC4**: Given that I copy a shape, Then all properties of the shape (e.g., size, position, fill colour, border) are preserved in the clipboard. **AC5**: Given that I copy a shape, Then the original shape remains fully intact and unmodified on the canvas. **AC6**: Given that multiple shapes are selected (if multi-selection is supported), When I click "Copy", Then all selected shapes are copied together to the clipboard.		
2.9	Paste operation	As a user, I want to be able to paste a shape into the drawing, So that I can reuse shapes within the drawing	**AC1**: Given that a shape has been previously cut or copied to the clipboard, When I click the "Paste" button, Then a copy of the shape is added to the canvas. **AC2**: Given that I paste a shape, Then it should appear in a default position (e.g., centered or slightly offset from the original if pasted multiple times). **AC3**: Given that the shape was copied, Then the original remains on the canvas, and the pasted shape is an identical duplicate. **AC4**: Given that the shape was cut, Then the pasted shape represents the moved object and retains all its original properties. **AC5**: Given that there is no shape in the clipboard, When I click "Paste", Then nothing happens, or an appropriate message is displayed (e.g., "Clipboard is empty"). **AC6**: Given that I paste a shape,	High	2

			Then all its properties (e.g., size, fill colour, border colour, type) are preserved in the new instance. **AC7**: Given that multiple shapes were copied or cut, When I click "Paste", Then all shapes are pasted together, maintaining their relative positions. **AC8**: Given that I paste a shape, Then I should be able to move, edit, or delete it like any other shape.		
2.10	Undo operation	As a user, I want to be able to undo an operation with no limit, So that I can restore the canvas to any previous state	**AC1**: Given that I have performed at least one operation, When I click the "Undo" button, Then the last operation is reverted, and the canvas is updated to reflect the previous state. **AC2**: Given that I have performed multiple operations, When I click "Undo" repeatedly, Then each previous state is restored step-by-step, with no limit on the number of undo levels. **AC3**: Given that I have not performed any operation, When I click "Undo", Then nothing happens, and the canvas remains unchanged. **AC4**: Given that I undo an operation, Then all properties affected by that operation (e.g., position, size, colour, presence on canvas) are accurately restored. **AC5**: Given that I undo a cut, delete, or move operation, Then the affected shape is returned to its original state and position on the canvas. **AC6**: Given that I undo a draw operation, Then the shape that was added is removed.	High	8
2.11	Bring to the front	As a user, I want to be able to bring a shape to the front, So that I can make an old shape more visible than newer ones	**AC1**: Given that there are multiple overlapping shapes on the canvas and one is selected, When I click the "Bring to the Front" button, Then the selected shape is moved above all other shapes, making it fully visible.	Medium-High	2

			AC2: Given that no shape is selected, When I click the "Bring to the Front" button, Then nothing happens. **AC3**: Given that the selected shape is already in the foreground, When I click "Bring to the Front", Then its position remains unchanged. **AC4**: Given that I bring a shape to the front, Then its position on the canvas (X/Y coordinates) remains unchanged — only its z-order is updated. **AC5**: Given that I undo the operation, Then the shape returns to its previous layering order. **AC6**: Given that multiple shapes are selected (if supported), When I click "Bring to the Front", Then all selected shapes are brought to the top, preserving their relative stacking order.		
2.12	Send to the back	As a user, I want to be able to send a shape to the back, So that I can make a new shape less visible than old ones	**AC1**: Given that there are multiple overlapping shapes on the canvas and one is selected, When I click the "Send to the Back" button, Then the selected shape is moved behind all other shapes, placing it at the bottom of the stacking order. **AC2**: Given that no shape is selected, When I click the "Send to the Back" button, Then nothing happens. **AC3**: Given that the selected shape is already at the back, When I click "Send to the Back", Then its position remains unchanged. **AC4**: Given that I send a shape to the back, Then its position on the canvas (X/Y coordinates) remains unchanged — only its z-order is updated. **AC5**: Given that I undo the operation, Then the shape returns to its previous layering order.	Medium-High	2

			AC6: Given that multiple shapes are selected (if supported), When I click "Send to the Back", Then all selected shapes are sent to the back together, preserving their relative stacking order.		
3.1	Zoom levels	As a user, I want to be able to zoom in and out of the canvas, So that I can better view specific part of the canvas	**AC1**: Given that the drawing area is displayed, When the user selects a different zoom level (e.g., via a slider, buttons, or dropdown), Then the entire drawing content scales up or down accordingly. **AC2**: Given that zooming occurs, Then the shapes and canvas content maintain their proportions and aspect ratio (no distortion). **AC3**: Given that the user zooms in, Then details become larger and easier to see, while the canvas area may become scrollable. **AC4**: Given that the user zooms out, Then the content becomes smaller, showing a larger portion of the drawing area. **AC5**: Given that a zoom level is applied, Then the position of shapes relative to each other remains unchanged. **AC6**: Given that the zoom level reaches minimum or maximum limits, Then further zooming in or out is disabled or prevented. **AC7**: Given that the user zooms, Then the zoom level control UI updates to reflect the current zoom percentage or factor.	Medium	3
3.2	Multiple predefined zoom levels	As a user, I want at least 4 different levels of zooming, So that I can easily zoom in and out of the canvas.	**AC1**: Given that the zoom control is available, When the user selects one of the predefined zoom levels (at least 4 levels, e.g., 50%, 100%, 150%, 200%), Then the drawing area adjusts to the selected zoom level. **AC2**: Given that the user selects a predefined zoom level, Then the drawing content scales proportionally without distortion. **AC3**: Given that a zoom level is selected,	Medium	3

			Then the zoom control visibly indicates the current zoom level. **AC4**: Given that the user changes zoom levels repeatedly, Then the zoom transitions smoothly and accurately to the selected level. **AC5**: Given that the user selects the same zoom level currently active, Then no unnecessary redraw or scaling occurs. **AC6**: Given that the user selects a zoom level outside the predefined options, Then the selection is either disallowed or snapped to the nearest available level.		
3.4	Scrollable Canvas	As a user, I want to scroll the drawing area when it's larger than the window, So that I can access hidden areas of the drawing	**AC1**. Given that I write drawing area width and height, When the the drawing area realizes a new width and/or height has been set, Then the drawing area changes its size according to the input set by the user; **AC2**. Given that the drawing area exceeds the visible window size, When the user scrolls horizontally or vertically, Then the viewport moves accordingly, revealing different parts of the drawing	Medium	3
3.5	Grid on / Grid off	As a user, I want to turn the background grid on or off, So that I can chose whether to use it or hide it	**AC1**: Given that a grid overlay is available, When the user enables the grid toggle, Then the grid is displayed on the background of the canvas. **AC2**: Given that the grid is currently visible, When the user disables the grid toggle, Then the grid is hidden from the canvas. **AC3**: Given that the grid is shown or hidden, Then the change is immediate and does not affect the existing shapes on the canvas. **AC4**: Given that the grid visibility has been changed, Then the new state (on/off) is retained until the user changes it again, even if switching tools.	Medium	2

			AC5: Given that I zoom in or out of the canvas, Then the grid (if enabled) scales proportionally to remain consistent with the zoom level. **AC6**: Given that the grid is visible, Then it does not obstruct or alter the drawing or selection of shapes.		
3.6	Adjustabl e Grid Size	As a user, I want to choose the spacing of the grid lines, So that I can easily change size	**AC1**: Given that the grid is currently visible, When the user changes the grid spacing setting via a slider, Then the canvas updates to show the new spacing between grid lines. **AC2**: Given that the user selects a larger grid size, Then the grid lines become more widely spaced on the canvas. **AC3**: Given that the user selects a smaller grid size, Then the grid lines become more densely spaced. **AC4**: Given that the grid spacing is changed, Then existing shapes and drawings on the canvas are not affected — only the grid appearance is updated. **AC5**: Given that the grid is not visible, When the user adjusts the grid spacing, Then the change is applied, and the updated spacing will be shown when the grid is made visible again. **AC6**: Given that the user zooms in or out, Then the adjusted grid spacing scales proportionally with the zoom level.	Medium	3
4.1	Drawing an irregular polygon	As a user, I want to draw a polygon that isn't regular, So that I'm not limited by default shapes	**AC1**: Given that I have selected the "polygon" tool, When I click in the drawing area, Then a vertex of the polygon is created at the cursor position. **AC2**: Given that I continue clicking in the drawing area, Then each new click adds a new vertex to the polygon. **AC3**: Given that I have added at least two vertices, Then I am able to draw a custom	Medium	8

			polygon. **AC4**: Given that I want to complete the polygon, When I click on the “Confirm” buttom, Then the polygon is closed, and the final shape is created. **AC5**: Given that I am drawing a polygon, When I press the Escape key, Then the drawing operation is cancelled, and no shape is created. **AC6**: Given that the polygon has been drawn, When I select it, I am able to perform some (but not all) operations on it.		
4.2	Text as shape	As a user, I want to write a text as a shape, So that I can further customize the drawing using elements that aren’t shapes	**AC1**: Given that I have clicked the “text” button, When I click on the drawing area, Then a text input area is created at the position of the cursor. **AC2**: Given that the text area is active, Then I can type and edit text directly within the shape. **AC3**: Given that I finish typing by pressing Enter, Then the text is rendered as a shape element on the canvas. **AC4**: Given that the text shape is selected, Then I can move, delete, or change its colour.	Medium	3
4.3	Rotate a shape	As a user, I want to rotate a shape of the drawing, So that I can easily edit it	**AC1**: Given that I have selected an existing shape on the canvas, When I move the “rotate” slider thumb, Then the shape is rotated by a default increment according to the value I selected. **AC2**: Given that the shape has been rotated, Then its position and proportions are preserved, but its orientation changes accordingly. **AC3**: Given that I repeatedly move the rotate slider thumb, Then the shape continues to rotate according to my selection. **AC4**: Given that the shape is rotated, Then it remains fully interactive, allowing me to move, resize, or recolour it afterward.	Medium	2

4.4	Mirror a shape	As a user, I want to mirror a shape of the drawing, both vertically and horizontally, So that I can easily edit it	**AC1**: Given that I have selected an existing shape on the canvas, When I click the "mirror" button, Then the shape is mirrored either vertically or horizontally depending on the selected option. **AC2**: Given that I click "mirror vertically," Then the shape is flipped along its vertical axis (left ↔ right). **AC3**: Given that I click "mirror horizontally," Then the shape is flipped along its horizontal axis (top ↔ bottom). **AC4**: Given that I want to undo the mirror operation, Then clicking the undo button restores the shape to its previous orientation. **AC5**: Given that I have multiple shapes selected, Then the entire group can be mirrored together.	Medium	2
4.5	Stretch a shape	As a user, I want to stretch a shape of the drawing, both vertically and horizontally, So that I can easily edit it	**AC1**: Given that I have selected an already drawn shape, When I click and drag a side border, Then the shape is stretched in that direction only (horizontally or vertically). **AC2**: Given that I click and drag a corner handle, Then the shape is stretched both horizontally and vertically, modifying its proportions. **AC3**: Given that I am dragging to stretch the shape, Then a live preview of the stretch is shown during the interaction. **AC4**: Given that I complete the stretching operation, Then the new shape dimensions are immediately applied and editable. **AC5**: Given that the shape includes text or grouped elements, Then all components are stretched proportionally along with the shape. **AC6**: Given that I press the Undo button after a stretch, Then the shape reverts to its original dimensions.	Medium	3
5.1	Group shape	As a user,	**AC1**: Given that I have selected two or more shapes,	Medium-Low	5

		I want to group two or more shapes in one, So that I can edit them at the same time	When I click the "group" button, Then the selected shapes are combined into one grouped shape. **AC2**: Given that the shapes are grouped, Then I can move, resize, rotate, delete, or style them at the same time. **AC3**: Given that I perform an operation (e.g., change color or stretch) on a group, Then the operation affects all shapes within the group simultaneously like it would if they were separated. **AC4**: Given that I select a grouped shape, Then the individual components are not directly editable unless ungrouped. **AC5**: Given that I undo a grouping operation, Then the shapes return to their previous ungrouped state.		
5.2	Ungroup shape	As a user, I want to ungroup a grouped shape, So that I can edit each shape separately	**AC1**: Given that I have selected a grouped shape, When I click on the "ungroup" button, Then the group is split into its original individual shapes. **AC2**: Given that the shapes are ungrouped, Then I can move, edit, delete, or style each shape independently. **AC3**: Given that the group was previously transformed (e.g., moved, resized), Then the individual shapes retain the final position, size, and styling applied while grouped. **AC4**: Given that the shape is not a group, Then I can't click the ungroup button. **AC5**: Given that I undo the ungroup operation, Then the shapes are restored to their grouped state. **AC6**: Given that I reselect some or all of the ungrouped shapes, Then I can reapply grouping as desired.	Medium-Low	3
5.3	Saving a group of shapes	As a user, I want to save grouped shapes as a custom shape,	Given that I have selected a grouped shape, When I click the "save custom shape" button,	Low	8

		So that I can immediately use that custom shape	Then the custom shape will be saved and ready to use		
5.4	Use grouped shapes	As a user, I want to use a custom shape previously saved, So that I don't have to re-create the custom shape	Given that I have selected a saved custom shape, When I click on the drawing area, Then the custom shape will be drawn into the canvas	Low	3
5.5	Save custom shape after the saving of the file	As a user, I want to save the drawing file, including any custom shape, So that I don't need to re-create the shape when I open the saved file	Given that I have drawn a custom shape, When I click on the "save file" button, Then the custom shapes will be saved alongside the file	Medium-Low	8
5.6	Export custom shape in a library	As a user, I want to export custom shapes to a library file, So that I can import all custom shapes whenever I want	Given that I have already saved custom shapes into the application, When I click on the "export to library" button, Then a new file containing custom shapes will be saved	Low	8
5.7	Import custom shape from a library	As a user, I want to import custom shapes from a library file, So that I can use all custom shapes I have previously created	Given that I have got an external library file, When I click on the "import from library", Then the custom shapes from the file will be available	Low	8

Definition of Done

The task can be considered done when:

- The code respects the convention that are defined by the team, and it doesn't introduce warnings.
- The code is tested with JUnit, and results are positive.

The user story can be considered done when:

- The Acceptance Criteria are satisfied.
- All the tasks of the User Story have been completed.

Language AND Development Environment

It will be used Java as programming language and JavaFX as framework for the graphical interface, using the MVC (Model-View-Controller) architecture.

As for development environment, IntelliJ will be the main IDE; other frameworks will be used to simplify development, such as SceneBuilder for GUI development.

There will be some conventions regarding operations, classes, variables, methods and other elements.

Naming convention:

- Variables and methods names will follow `camelCase` convention (e.g. `lineSegment`, `drawLineSegment()`, etc.).
- Classes and interfaces will follow `UpperCamelCase` convention (e.g. `Rectangle.java`).
- Constants will follow `UPPER_SNAKE_CASE` convention (e.g. `MAX_SHAPE_NUMBERS`).
- Files and resources (such as images) will follow `kebab-case` convention (e.g. `undo-icon.svg`).
- Folders and packages will use lowercase with no special character (e.g. `model/`, `view/`, `controller/`).

Other conventions:

- Comments are mandatory, they must be written in English and must be not redundant.
- Code must be formatted.
- Classes, methods, variables and packages' names won't be excessively long.
- Classes, methods, and variables and packages' names will be clear.
- Variables will follow naming conventions, such as:
  - GUI elements' name will be composed of their functionality + the type of the GUI node (e. g., the button that draws a rectangle will be called `drawRectangleButton`).
  - The drawing area will be referred to as "Canvas".
- Recurring operations that compose methods' names will follow naming conventions (e. g., "to insert a shape into the drawing area" will be known as "drawing on canvas"; so, a method whose function is to add a rectangle to the drawing aura will be called `drawRectangle` (for simplicity "on canvas" will be omitted).

Glossary

	For the user	For the developer
Canvas	For the user, a canvas is the drawing area	For the developer, a canvas is a component that contains shapes.
Draw	For the user, as the word suggests, drawing means creating a shape on the canvas	For the developer, drawing means creating and adding shapes components in the View. In the Model, for manipulating shapes data, common words used for collections will be used – e.g. add a shape to a map.
Shape	For the user, a graphical element that can be drawn or that can be manipulated (moved, resized, etc.)	For the developer, in the View, a shape is a JavaFX node, child of the canvas. In the Model, a shape is a class that contains all the logic related to a shape (position, width, length, centers, radii, foci, etc.)
Selection	For the user, selection is the action that lets the user click a shape and then manipulate it.	For the developer, selection is the action that marks a shape data (in the model). In the view, this selection mark is used to give an effect to existing shapes.
Preview	For the user, a preview is either a draft of a drawing that is yet to be drawn or a drawing that's being modified.	For the developer, a preview is a temporary shape that is used by the user (it is used by the View and it doesn't exist in the Model).
Model	*(this term isn't known by the user)*	The Model is the MVC layer that manages logic. It is invisible to the user.
View		The View is the MVC layer that is viewed by the user.
Controller		The Controller is the MVC layer that acts as a bridge between Model and View.

SPRINT PLANNING

FIRST SPRINT

Team velocity (ideal): 47 story points, but we are estimated 31 Story Points for the first Sprint, even if velocity isn't ideal. As this is our first time working together, we decided to intentionally set a lower velocity estimate to avoid overcommitting.

ID	TITLE	TASK	ASSIGNED TO	SP
1.1	Empty window on start-up	Create JavaFX project	Mario Izzo	1
		Create method start in main		
		Create `Canvas` class		
		Add graphical elements to GUI		
1.2	Line drawing	Create `Line` class	Sabato Paolillo	2
		Create `line` button	Alessandro Maruotto	
		Create method `drawLine`	Pierluigi Pio Nocerino	
		Bind `line` button to method	Mario Izzo	
1.3	Rectangle drawing	Create `Rectangle` class	Sabato Paolillo	2
		Create `Rectangle` button	Alessandro Maruotto	
		Create method `drawRectangle`	Pierluigi Pio Nocerino	
		Bind `rectangle` button to method	Mario Izzo	
1.4	Ellipse drawing	Create `Ellipse` class	Sabato Paolillo	2
		Create `Ellipse` button	Alessandro Maruotto	
		Create method `drawEllipse`	Pierluigi Pio Nocerino	
		Bind `ellipse` button to method	Mario Izzo	
1.5	Shape Color	Add a colour palette panel	Mario Izzo	3
		Create a method `setShapeFillColor`	Pierluigi Pio Nocerino	
		Create abstract `Shape` class and make `Line`, `Rectangle` and `Ellipse` classes extend from `Shape`	Sabato Paolillo	
		Update method `drawEllipse` to draw from abstract Shape class	Alessandro Marutto	
		Update method `drawRectangle` to `draw` from abstract `Shape` class		
		Update method `drawLine` to `draw` from abstract `Shape` class		
1.6	Border Colour	Add another colour palette	Mario Izzo	2
		Create a method `setBorderFillColor`	Pierluigi Pio Nocerino	
		Update method `draw` in `Ellipse` class		

		Update method `draw` in `Rectangle` class	Alessandro Maruotto	
		Update method `draw` in `Line` class		
1.7	Saving the drawing in a file	Add `save drawing` button	Sabato Paolillo	8
		Create method that exports drawing to file	Pierluigi Pio Nocerino	
		Bind method to button	Mario Izzo	
1.8	Open a drawing via file	Add `load drawing` button	Mario Izzo	8
		Create method that opens a file and reads it	Pierluigi Pio Nocerino	
		Add the drawing in the canvas	Alessandro Maruotto	
2.1	Shape of the drawing	Add `isSelected` property to Shape class	Sabato Paolillo	1
		Create method `handleMouseClick`	Pierluigi Pio Nocerino	
		Highlight selected shape	Mario Izzo	
		Deselect previously selected shape when another is clicked	Pierluigi Pio Nocerino	
2.2	Delete Operation	Create `delete` button	Mario Izzo	2
		Create method `deleteShape` in `Canvas`	Pierluigi Pio Nocerino	
		Bind method to button	Alessandro Maruotto	

SECOND SPRINT

Team ideal velocity: 56 Story Points. For Sprint 2, we are estimating 45 Story Points, which, while not our ideal velocity, represents a planned increase. In Sprint 1, we intentionally set a lower velocity estimate (31 SP) to establish a baseline and avoid overcommitment as a new team. We successfully completed all planned tasks for Sprint 1. Given this positive outcome and our improved understanding of our collective capacity, we have decided to cautiously increase our velocity estimate for Sprint 2 to 45 SP. This increment reflects our growing confidence and aims to progressively move us closer to our ideal team velocity.

ID	TITLE	TASK	ASSIGNED TO	SP
2.3	Shape Preview	Update `isSelected` flag in `ShapeData` via `CanvasModel` when a shape is clicked.	Mario Izzo	3
		Clear `isSelected` flag for all `ShapeData` in `CanvasModel` when canvas background is clicked.	Alessandro Maruotto	
		Add mouse event handling in `Controller.setOnMousePressed()` to identify if a click is on a shape.	Pierluigi Pio Nocerino	
		Trigger `SelectShapeCommand` in `Controller` on shape click.	Mario Izzo	
		Trigger `DeselectAllShapeCommand` in `Controller` on canvas background click.	Sabato Paolillo	
		Apply/remove visual selection highlight in `CanvasView` via `Highlighter`	Pierluigi Pio Nocerino	
2.4	Move Operaration	Implement mouse drag detection on a selected shape in `Controller`.	Sabato Paolillo	5
		Update `ShapeData` coordinates in `CanvasModel` during drag via `Controller`	Alessandro Maruotto	
		Ensure continuous visual update of the shape's position in `CanvasView` during drag.	Pierluigi Pio Nocerino	
		Create and execute `MoveShapeCommand` on mouse release to finalize the new position in `CanvasModel`.	Mario Izzo	
		Add boundary checks in `Controller` or `CanvasModel` to prevent dragging shapes off-canvas.	Mario Izzo	
		Verify shape's visual style remains unchanged after move operation during `CanvasView` repaint.	Sabato Paolillo	
		Ensure drag operations do not move shapes if no shape is selected.	Alessandro Maruotto	
2.5	Changing colour operation	Add method in `CanvasModel` to update fill color for selected shape(s).	Sabato Paolillo	2
		Add method in `CanvasModel` to update stroke/border color for selected shape(s).	Alessandro Maruotto	
		Create `ChangeShapeFillColorCommand` to encapsulate fill color changes in `CanvasModel`.	Pierluigi Pio Nocerino	
		Create `ChangeShapeStrokeColorCommand` to encapsulate stroke color changes in `CanvasModel`.	Mario Izzo	
		Connect `fillColorPicker` action in `Controller` to create and execute	Sabato Paolillo	

		`ChangeShapeFillColorCommand` for selected shape(s)		
		Connect `strokeColorPicker` action in `Controller` to create and execute `ChangeShapeStrokeColorCommand` for selected shape(s).	Pierluigi Pio Nocerino	
		Add logic in `Controller` to prevent color change and optionally display a message if no shape is selected when a color picker is used.	Mario Izzo	
		Ensure `CanvasModel` updates (fill or stroke) trigger a `CanvasView` repaint reflecting the new color(s) while retaining other properties.	Pierluigi Pio Nocerino	
2.6	Changing size operation	Implement visual resize handles in `CanvasView` that appear around a selected shape.	Alessandro Maruotto	8
		Add logic in `Controller` to detect mouse press and drag operations on the resize handles.	Alessandro Maruotto	
		Add method(s) in `CanvasModel` to update dimensions of a `ShapeData` object based on handle drag.	Pierluigi Pio Nocerino	
		Implement logic in `Controller` or `CanvasModel` for free resizing (side/top/bottom handles).	Mario Izzo	
		Implement logic in `Controller` or `CanvasModel` for aspect-ratio-locked resizing (corner handles).	Mario Izzo	
		Ensure `CanvasView` provides real-time visual feedback of resizing during drag.	Sabato Paolillo	
		Create `ResizeShapeCommand` to finalize shape dimensions in `CanvasModel` upon mouse release.	Alessandro Maruotto	
		Verify that other shape properties (color, style, etc.) remain unchanged in `ShapeData` and `CanvasView` after resizing.	Mario Izzo	
		Ensure no resize operation occurs if no shape is selected or if drag is not on a resize handle.	Pierluigi Pio Nocerino	
2.7	Cut operation	Implement an internal clipboard mechanism (e.g., a field in `CanvasModel` or a new dedicated class) to temporarily store `ShapeData`.	Mario Izzo	2
		Add method in `CanvasModel` to store a deep copy of the selected `ShapeData` (with all its properties) into the internal clipboard.	Pierluigi Pio Nocerino	
		Add method in `CanvasModel` to remove the selected shape(s) from the canvas (similar to `deleteShapes` but targeted by cut operation).	Sabato Paolillo	
		Create `CutShapeCommand` that copies the selected shape's data to the clipboard and then removes the shape from the `CanvasModel`.	Alessandro Marutto	
		Add a "Cut" button to the UI (FXML) and connect its action in `Controller` to create and execute the `CutShapeCommand`.	Sabato Paolillo	
		Implement logic in `Controller` or `CutShapeCommand` to do nothing if no shape is selected when "Cut" is triggered.	Pierluigi Pio Nocerino	
		Ensure `CanvasView` is updated to reflect the removal of the cut shape.	Mario Izzo	

2.8	Copy operation	Adapt or ensure the internal clipboard mechanism (e.g., in `CanvasModel`) can store one or more `ShapeData` objects (deep copies).	Mario Izzo	2
		Add method in `CanvasModel` to retrieve deep copies of all currently selected `ShapeData` objects, preserving all their properties.	Pierluigi Pio Nocerino	
		Create `CopyShapeCommand` that retrieves copies of selected shape(s) data from `CanvasModel` and stores them in the internal clipboard.	Alessandro Maruotto	
		Add a "Copy" button to the UI (FXML) and connect its action in `Controller` to create and execute the `CopyShapeCommand`.	Sabato Paolillo	
		Implement logic in `Controller` or `CopyShapeCommand` to do nothing if no shape is selected when "Copy" is triggered.	Mario Izzo	
		Ensure the original shape(s) on the canvas remain unchanged in `CanvasModel` and `CanvasView` after the copy operation.	Pierluigi Pio Nocerino	
2.9	Paste operation	Add method in `CanvasModel` to retrieve deep copies of `ShapeData` object(s) currently stored in the internal clipboard.	Sabato Paolillo	2
		Implement logic in `CanvasModel` to add these new `ShapeData` instances to the canvas, assigning new unique IDs and calculating default positions (e.g., offset from original, or centered).	Pierluigi Pio Nocerino	
		Create `PasteShapeCommand` that retrieves shape data from clipboard and instructs `CanvasModel` to add them as new shapes.	Mario Izzo	
		Add a "Paste" button to the UI (FXML) and connect its action in `Controller` to create and execute the `PasteShapeCommand`.	Sabato Paolillo	
		Implement logic in `Controller` or `PasteShapeCommand` to do nothing (or show a message) if the clipboard is empty when "Paste" is triggered.	Mario Izzo	
		Ensure `CanvasView` is updated to render the newly pasted shape(s).	Alessandro Maruotto	
		Verify that pasted shapes are new, independent instances that can be selected, moved, edited, and deleted like any other shape on the canvas.	Pierluigi Pio Nocerino	
2.10	Undo operation	Add void `undo()` method definition to the `Command` interface.	Mario Izzo	8
		Implement an undo stack (e.g., `Stack<Command>`) in `CommandManager` to store executed commands.	Pierluigi Pio Nocerino	
		`CommandManager.executeCommand(Command command)` to push the command onto the undo stack after successful execution.	Alessandro Maruotto	
		Implement `CommandManager.undo()` method to pop the last command from the stack and call its `undo()` method.	Sabato Paolillo	
		Add an "Undo" button to the UI (FXML) and connect its action in `Controller` to call `commandManager.undo()`.	Mario Izzo	

		Ensure `CommandManager.undo()` does nothing if the undo stack is empty.		
		Ensure `CanvasModel.notifyObservers()` is called after an `undo()` operation modifies the model, so `CanvasView` updates.		
2.11	Bring to the front	Add method in `CanvasModel` to change the z-order of selected shape(s) by moving their `ShapeData` entries to the end of the internal `LinkedHashMap` (which dictates rendering order).	Sabato Paolillo	2
		Create `BringToFrontCommand` that instructs `CanvasModel` to reorder the selected shape(s). This command must store the original z-order (e.g., original map iteration indices or relative order of affected shapes) for undo.	Pierluigi Pio Nocerino	
		Add a "Bring to the Front" button to the UI (FXML) and connect its action in `Controller` to create and execute `BringToFrontCommand`.	Mario Izzo	
		Implement logic in `Controller` or `BringToFrontCommand` to do nothing if no shape is selected.	Pierluigi Pio Nocerino	
		Add logic in `CanvasModel` or `BringToFrontCommand` to handle cases where the shape is already at the front (no change needed, or command doesn't execute).	Alessandro Maruotto	
		Ensure `BringToFrontCommand.execute()` only changes z-order and not X/Y coordinates or other properties.	Alessandro Maruotto	
		Implement `BringToFrontCommand.undo()` to restore the shape(s) to their previous z-order in `CanvasModel`.		
		Ensure `CanvasModel.notifyObservers()` is called after z-order changes (execute or undo) so `CanvasView` re-renders shapes in the new order.	Pierluigi Pio Nocerino	
2.12	Send to the back	Add method in `CanvasModel` to change the z-order of selected shape(s) by moving their `ShapeData` entries to the beginning of the internal `LinkedHashMap` (which dictates rendering order).	Mario Izzo	2
		Create `SendToBackCommand` that instructs `CanvasModel` to reorder the selected shape(s) to the bottom of the z-order. This command must store the original z-order (e.g., original map iteration indices or relative order of affected shapes) for undo.	Pierluigi Pio Nocerino	
		Add a "Send to the Back" button to the UI (FXML) and connect its action in `Controller` to create and execute `SendToBackCommand`	Alessandro Maruotto	
		Implement logic in `Controller` or `SendToBackCommand` to do nothing if no shape is selected.	Sabato Paolillo	
		Add logic in `CanvasModel` or `SendToBackCommand` to handle cases where the	Mario Izzo	

		shape is already at the back (no change needed, or command doesn't execute).		
		Ensure `SendToBackCommand.execute()` only changes z-order and not X/Y coordinates or other properties.	Sabato Paolillo	
		Implement `SendToBackCommand.undo()` to restore the shape(s) to their previous z-order in `CanvasModel`.	Alessandro Maruotto	
		Ensure `CanvasModel.notifyObservers()` is called after z-order changes (execute or undo) so `CanvasView` re-renders shapes in the new order.	Pierluigi Pio Nocerino	
3.1	Zoom levels	Add a GUI element that lets user select the zoom level	Mario Izzo	3
		Bind zoom logic to the zoom component	Mario Izzo	
3.2	Multiplepredefined zoom levels	Add a GUI element that allows user to select zoom factor	Sabato Paolillo	3

Third Sprint

Team ideal velocity: 67 Story Points. For Sprint 3, we are estimating 35 Story Points, which, while not our ideal velocity, represents a planned increase. In Sprint 1, we intentionally set a lower velocity estimate (31 SP) to establish a baseline and avoid overcommitment as a new team. We successfully completed all planned tasks for Sprint 1. Given this positive outcome and our improved understanding of our collective capacity, we have decided to cautiously increase our velocity estimate for Sprint 2 to 45 SP. This increment reflects our growing confidence and aims to progressively move us closer to our ideal team velocity. Considering that all the logic made by creating methods, commands, etc. (and so by completing tasks) is usually changed later, we decided to make tasks more generic.

ID	Title	Task	Assigned to	SP
3.5	Grid on / Grid off	Add grid button for the GUI	Alessandro Maruotto	2
		Create method(s) that manages grid being visible	Pierluigi Pio Nocerino	
		Bind method to button	Sabato Paolillo	
		Make grid size match canvas size	Alessandro Maruotto	
		Make grid size match canvas zoom	Alessandro Maruotto	
3.6	Adjustable Grid Size	Add a slider for grid spacing	Sabato Paolillo	3
		Create methods that manage grid drawing in consideration of grid spacing	Alessandro Maruotto	
		Bind slider to method, so that changes are immediately visible when changing slider value (or when grid button is selected)	Sabato Paolillo	
		Make grid not affect other elements	Sabato Paolillo	
4.1	Drawing an irregular polygon	Add a draw polygon button on the GUI	Mario Izzo	8
		Create a new state for drawing a custom polygon	Mario Izzo	
		Create methods that manages vertices logic	Alessandro Maruotto	
		Create GUI element that lets user confirm their vertices	Pierluigi Pio Nocerino	
		Create methods (and commands) that connect vertices and draw a custom polygon	Sabato Paolillo	
4.2	Text as a shape	Add Text button to the GUI	Pierluigi Pio Nocerino	3
		Create methods, commands and classes that manage text logic	Pierluigi Pio Nocerino	

		Add method that manages text input logic	Pierluigi Pio Nocerino	
4.3	Rotate a shape	Add a circular slider to the GUI	Mario Izzo	2
		Create method that manages shape rotation preview	Mario Izzo	
		Create methods that manage shape rotation logic	Alessandro Maruotto	
		Bind methods to slider so that changes are visible immediately	Alessandro Maruotto	
4.4	Mirror a Shape	Add mirror buttons to GUI	Mario Izzo	2
		Create methods and commands that manage shape mirroring	Mario Izzo	
5.1	Group Shape	Add a group button to the GUI	Sabato Paolillo	5
		Create methods, commands and classes that manage grouping logic	Mario Izzo	
		Make button enabled only if at least 2 shapes are selected	Pierluigi Pio Nocerino	
		Create methods that manage grouped shapes operations	Pierluigi Pio Nocerino	
5.2	Ungroup shape	Add ungroup button to the GUI	Pierluigi Pio Nocerino	3
		Create methods and commands that manage ungrouping logic	Alessandro Maruotto	
		Make button not selectable when a grouped shape is not selected	Mario Izzo	

Implemented Pattern

Type	Role	Actor
Factory Pattern		
It is used to manage Shapes Data creation		
`Interface`	`Product`	`ShapeData`
`Classes` (they implement `Product`)	`Concrete Product`	`LineData`
		`EllipseData`
		`RectangleData`
		…
`Interface`	`Creator`	`ShapeFactory`
`Classes` (they implement `Creator`)	`Concrete Creator`	`LineFactory`
		`EllipseFactory`
		`RectangleFactory`
		…
`Class`	`Client`	`Controller`, while in drawing state
Composite Pattern		
It is used to treat individual shapes and composed shapes uniformly		
`Abstract Class`	`Component`	`ShapeData`
`Classes`	`Leaf`	`LineData`

		EllipseData
		RectangleData
		…
Class (it extends Component)	Composite	GroupedShapeData
Class	Client	CanvasModel
COMMAND PATTERN		
It is used to contain all the information about an action		
Interface	Command	Command
Classes (they implement Command)	Concrete Command	AddShapeCommand
		DeleteShapeCommand
		…
Class	Receiver	CanvasModel
Class	Invoker	CommandManager
Class	Client	Controller
OBSERVER PATTERN		
It is used to allow the Model notify the View		
Interface	Observer/Subscriber	ModelObserver
Classes	Concrete Observer/Concrete Subscriber	Controller
		SelectionPropertyObserver (disables GUI elements when shapes aren't selected)
Class	Subject/Publisher	CanvasModel
STATE PATTERN		
It is used to manage the way the application reacts to the user input		
Interface	State	State
Classes (they implement State)	Concrete States	IdleState
		SelectionState
		DrawingState
		MovingState
		ResizingState
		…
Class	Context	Controller (changes state based on the selected button)
ADAPTER PATTERN		
It is used to convert data from Model to View		
Object	Target	JavaFX Shape
Class	Client	Controller
Interface and classes which implement it	Adaptee	ShapeData
		LineData
		…
Class	Adapter	LineAdapter
		EllipseAdapter
		RectangleAdapter

Class (it isn’t strictly part of the pattern)	Helper	ShapeConverter