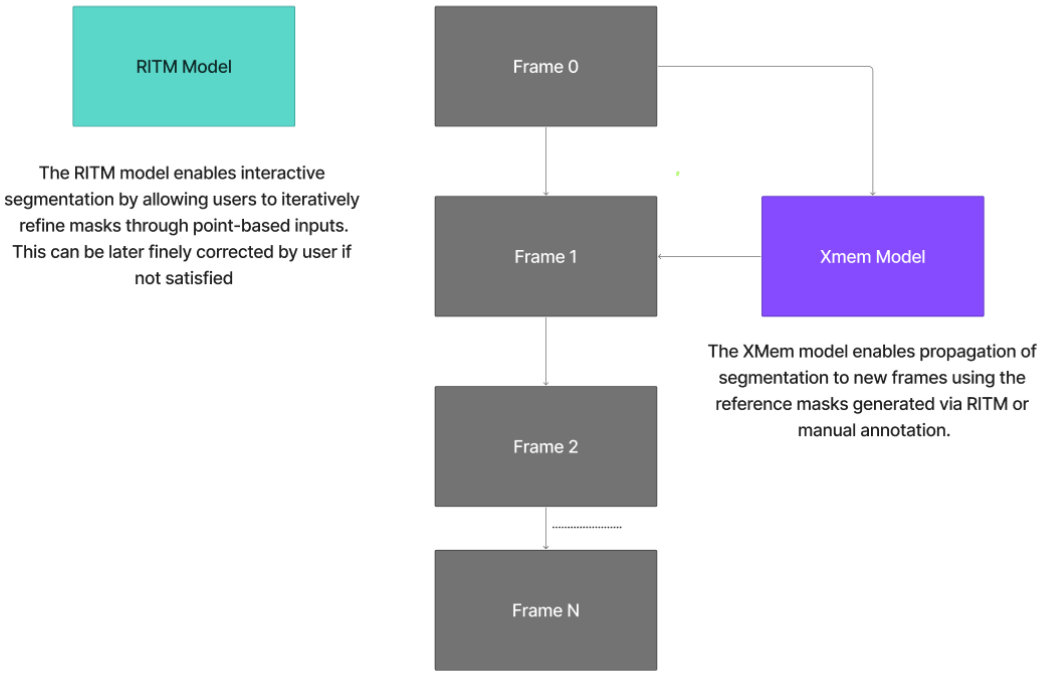
Samsung Prism: Work-let Area – Computer Graphics

A software tool to generate Multi-view Image Segmentation & Correction

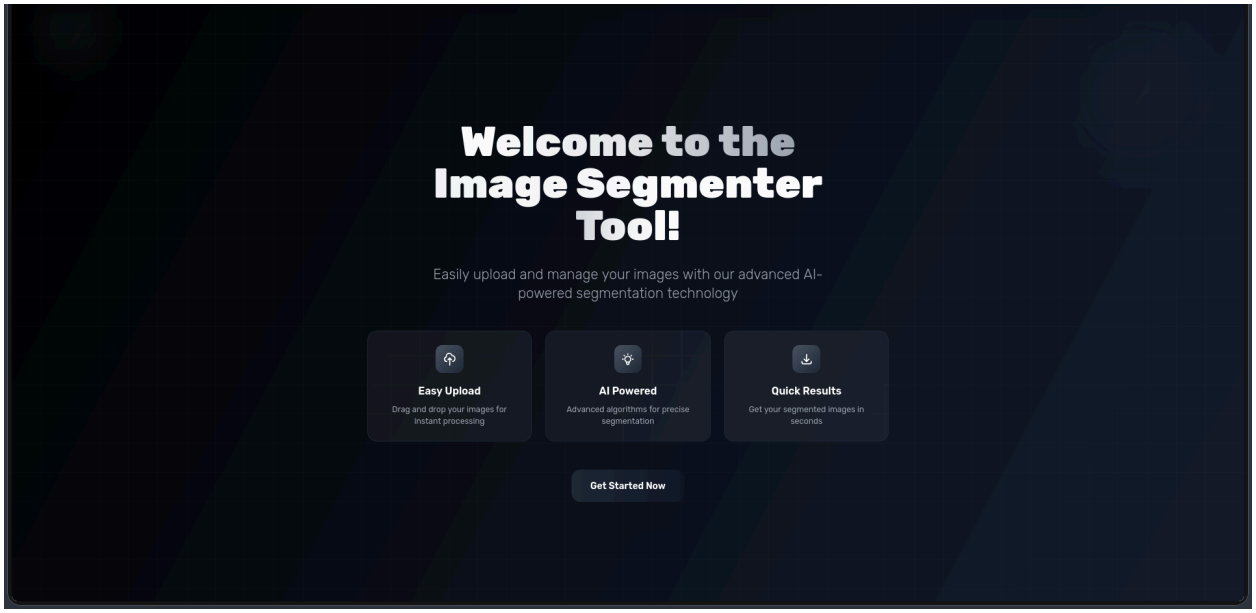
This tool is a web-based platform designed for interactive image segmentation, annotation. It integrates deep learning models RITM and XMem within an intuitive user interface, supporting workflows ranging from image-level annotation to long-range video propagation.



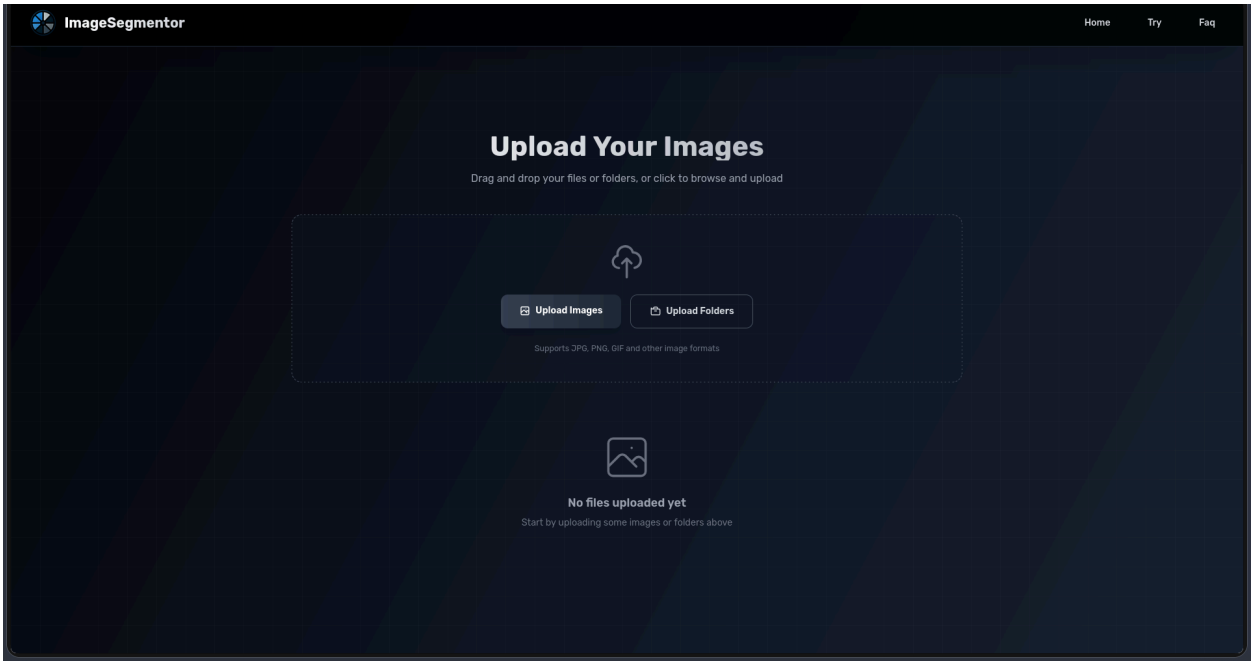
Overall Architecture

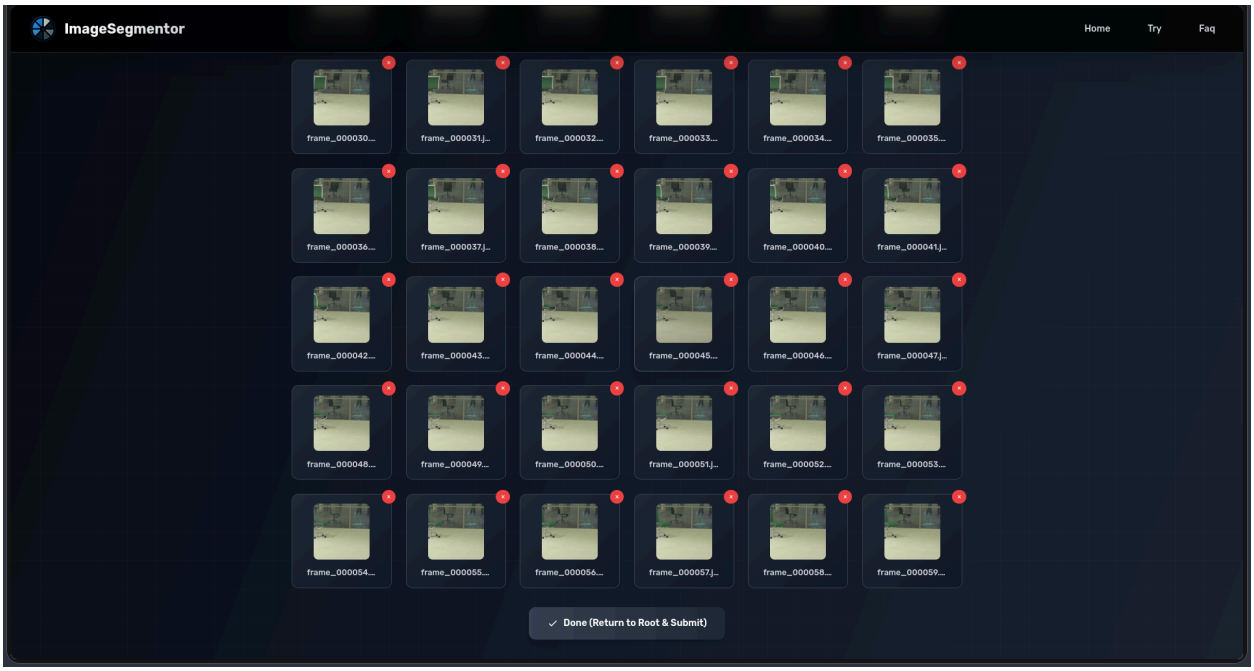
Overall Flow and User Interaction

Upon launching the tool, users are presented with a clean landing page featuring a navigation bar. The navigation bar includes links to the **Home**, **Try**, and **FAQ** sections. Clicking on these buttons either scrolls to the relevant section or navigates the user accordingly.



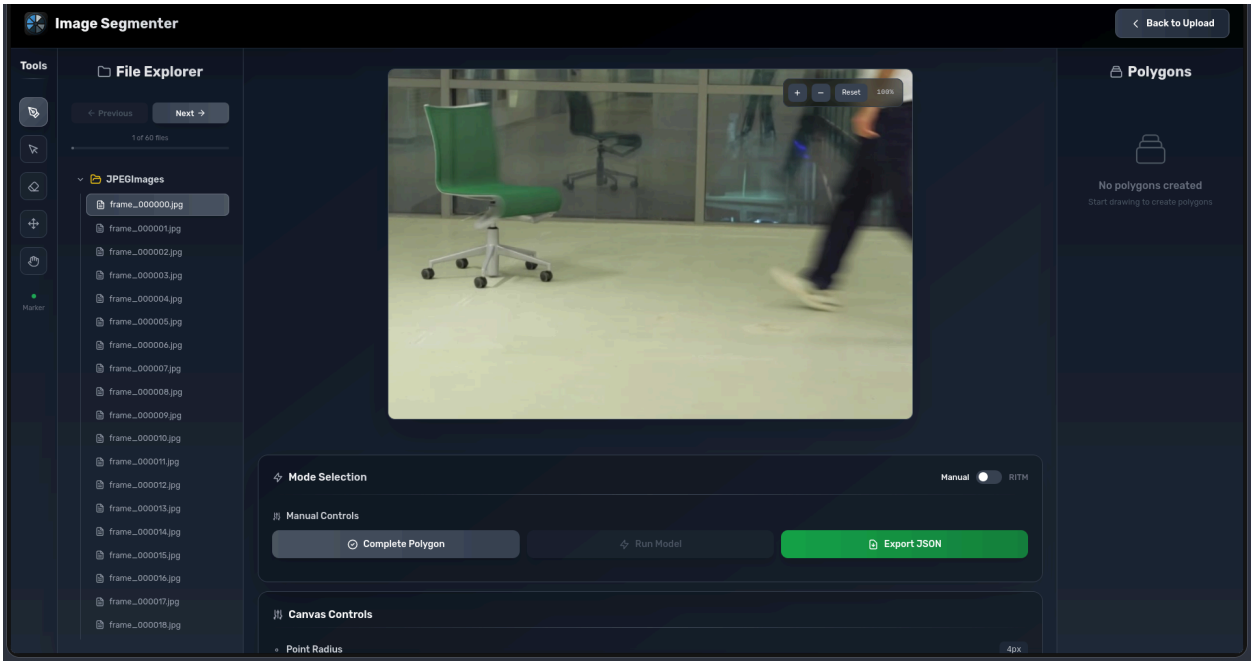
In the **Try** section, users can begin annotating by uploading **either a folder containing multiple images or individual image files.** The interface is designed to be simple yet powerful: users can switch between uploaded images or folders, delete unnecessary items using a convenient delete icon, or upload additional files as needed. This flexible upload and preview system streamlines the preparation process for annotation.





Segmentation Workflow and Features

Once the images are uploaded, users are directed to the **Preview Section**, where they can begin interacting with their data. Here, several tools are available to convert between polygon and mask representations. For instance, the **Convert to Mask** button transforms an editable polygon into a static mask, whereas the **Convert to Polygon** button allows previously masked regions to be re-edited by converting them back into point-based polygons.



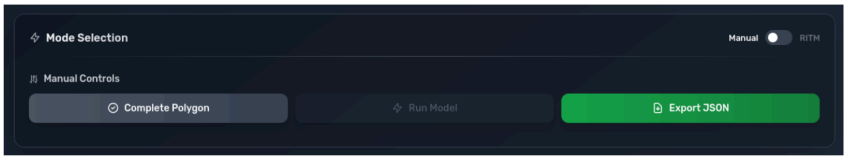
To further customize the annotation process, the tool includes a **Point Count Slider** that adjusts the number of automatically generated points along the polygon boundary. Importantly, this count cannot be set below the number of manually added points. A **Point Radius Slider** also enables users to change the visual size of points on the canvas, improving clarity during detailed edits. Once users are satisfied with their initial annotations, clicking **Done** transitions them into the **Editing Screen**.

Editing Screen Structure

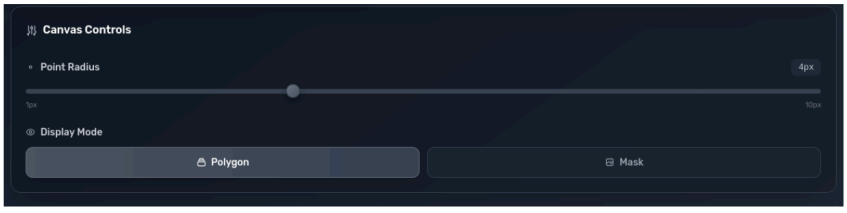
The Editing Screen is organized into four main panels: the **Toolbar**, **Folder Tree**, **Preview Section**, and **Polygon Tray**.

The **Toolbar** provides essential tools for polygon editing. The **Marker Tool** is used to place points and form polygons. The **Move Tool** allows users to reposition individual points or add new ones by clicking on the polygon edge. The **Eraser Tool** deletes selected points, and the **Move Polygon Tool** lets users reposition entire polygon structures.

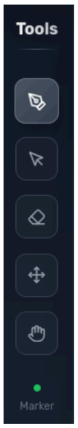
The **Folder Tree** displays the full image hierarchy that was uploaded. Users can easily navigate between images by clicking on filenames and using **Previous** and **Next** buttons for sequential browsing.



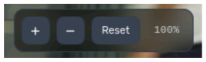
Toggle lets you choose between Manual Editing and RITMs interactive Correction



This pane lets you change point density and point radius and also switch between polygon and mask view



1) Add Point
2) Select and Move points
3) Erase Point
4) Move Mask
5) Navigate through IMage



Zoom Pane

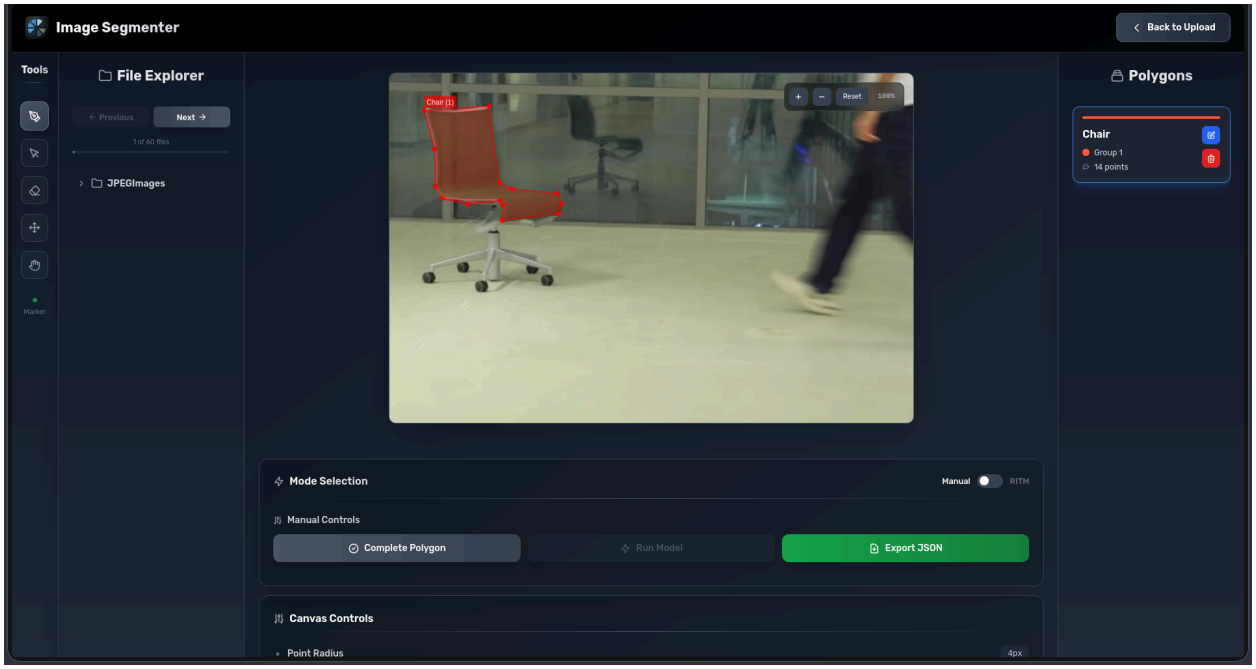
Samsung Research

The **Preview Section** is the main canvas for segmentation. It supports dynamic interactions such as zooming and panning, real-time polygon edits, and toggling between polygon and mask views. The canvas integrates tightly with the polygon and point controls, allowing fine-grained edits with immediate visual feedback.

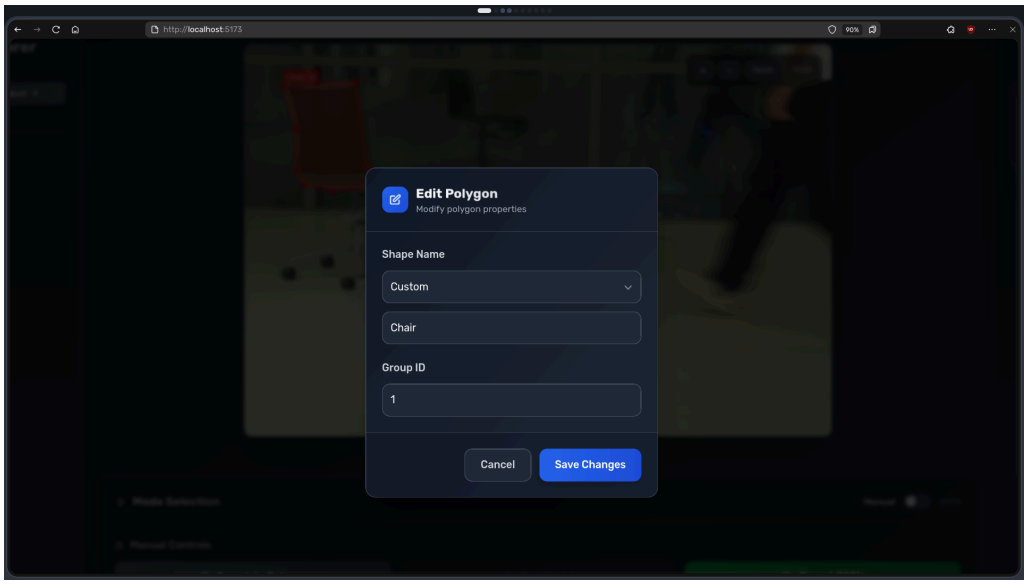
The **Polygon Tray** maintains a list of all polygons associated with the currently selected image. Clicking on any polygon entry highlights it on the canvas. Users can delete, edit, or convert individual polygons through contextual controls. The tray is synchronized in real-time with the canvas, ensuring changes are reflected instantly.

Labeling, Grouping, and Metadata

When a polygon is closed, a dialog prompts the user to assign a **name** and an optional **group ID** to it. Users can choose from a predefined list of labels or create new ones that are stored for reuse. Grouping polygons is useful for categorizing instances, managing sequences, or preparing structured training datasets.



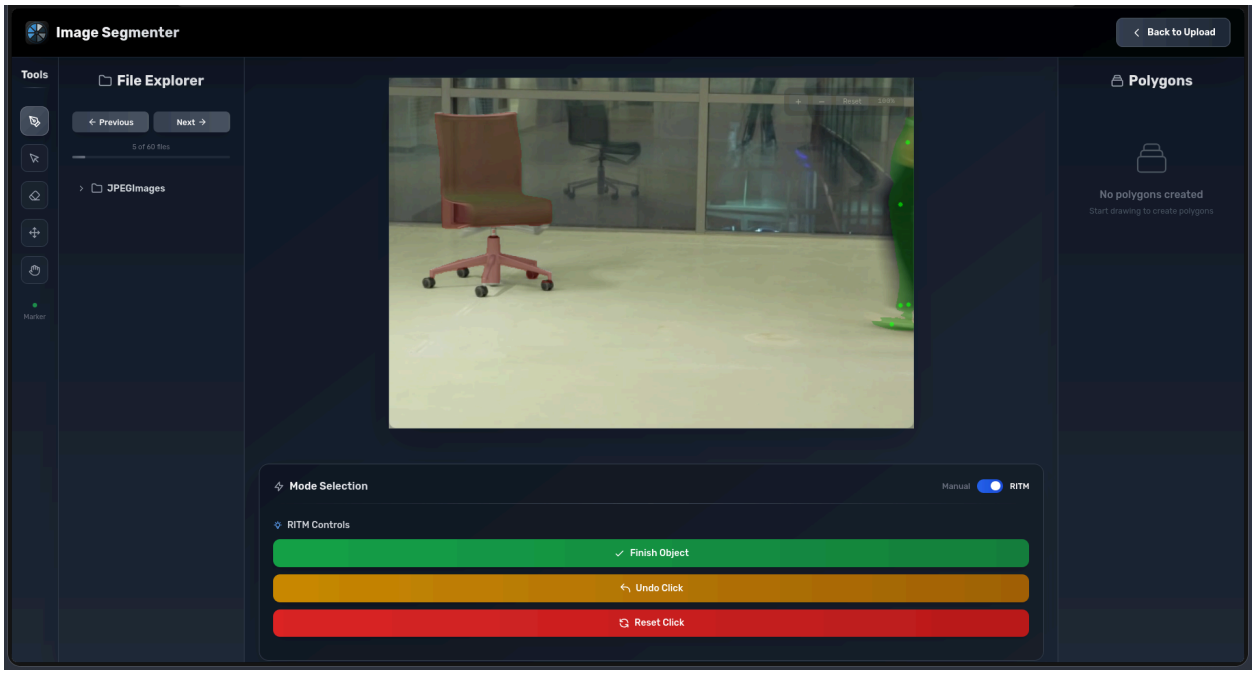
Here you can edit the Class Objects and Instances



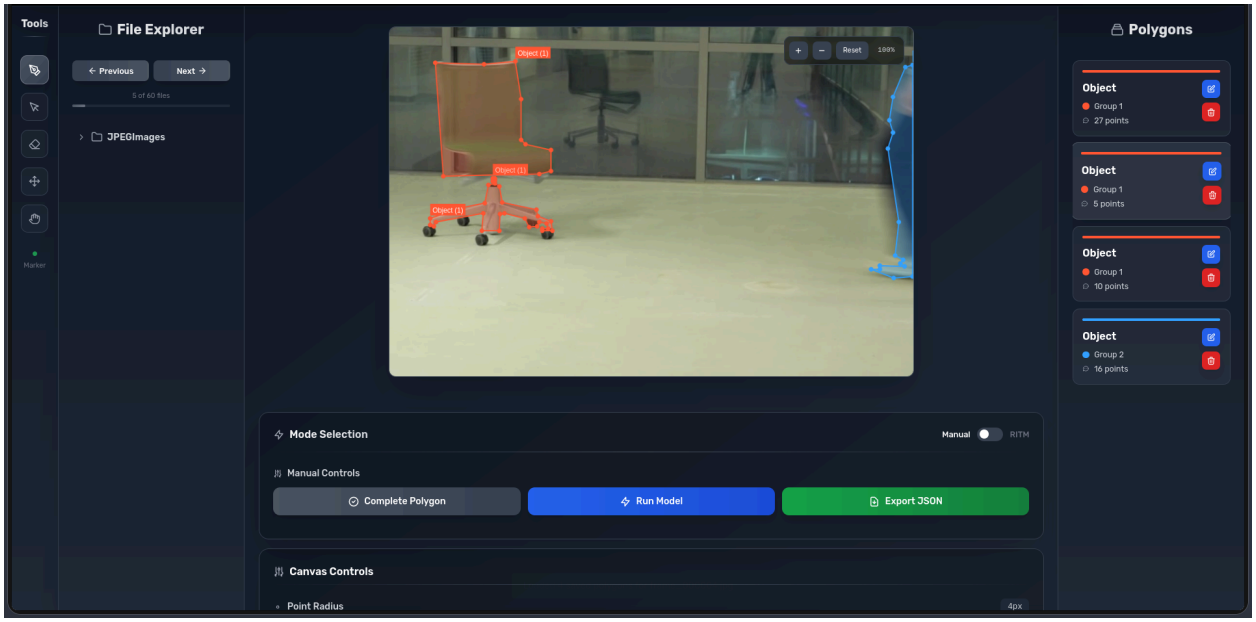
Deep Learning Integration

The tool supports both manual and AI-assisted segmentation workflows. Using the **RITM model (Reviving Iterative Training with Mask guidance)**, users can perform point-based interactive segmentation. By clicking on foreground and background areas, users iteratively refine the segmentation mask. This feature is ideal for quickly generating high-quality reference frames.

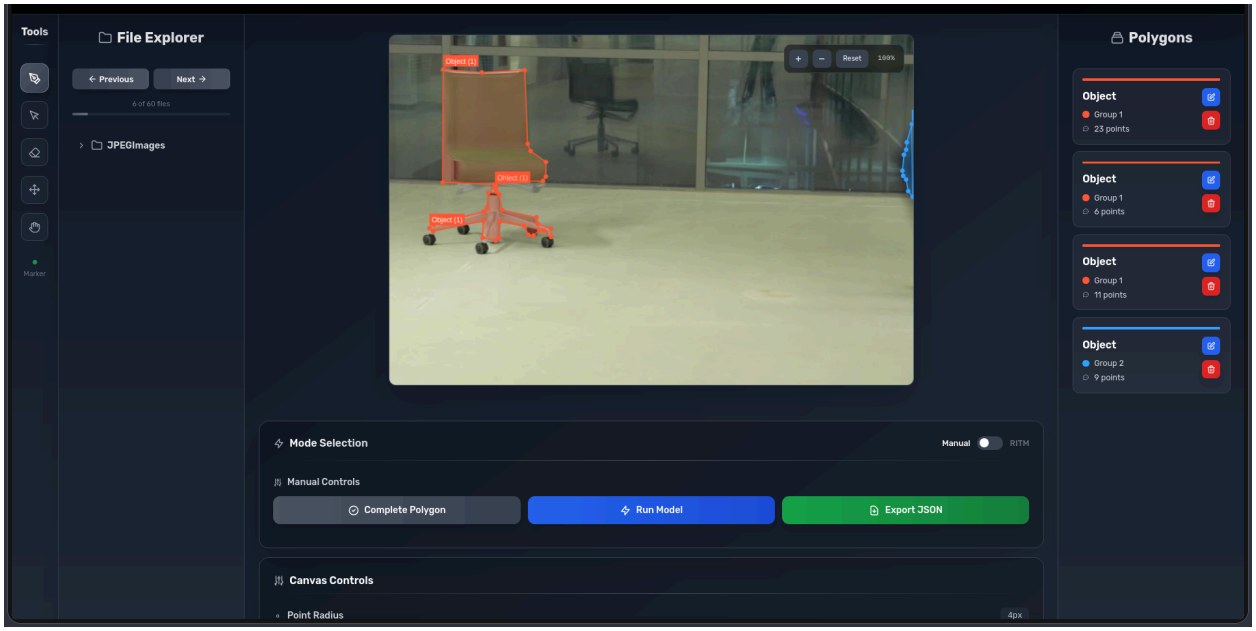
For video or sequence-based workflows, the **XMem model** is integrated to propagate annotated frames forward in time. Once a reference mask (from RITM or manual annotation) is defined, XMem automatically tracks and segments the object across future frames. This dramatically reduces manual effort when dealing with long videos.



Left click on the canvas to get get RITMs maks outputs, (this takes a couple seconds to render)



Once you toggle back to Manual, you can edit the masks generated manually



This is the propagated output once Run Model is pressed and the xmem output is converted to a json and rendered.

Technology Stack

The tool's frontend is built entirely in **React.js**, offering a fast, modular, and maintainable architecture. React's component-based design enabled the development of isolated and reusable UI elements such as the toolbar, preview canvas, and polygon tray. Real-time interactivity is powered by React's efficient state management using hooks like `useState` and `useReducer`, while the virtual DOM ensures smooth re-rendering for a responsive editing experience.

Several specialized libraries are integrated to enhance functionality. Canvas-based drawing is supported through `fabric.js` or `konva`, file handling is managed via `react-dropzone`, and navigation is handled with `react-router`. Optional state management can be extended using libraries like `redux` or `zustand`.

The backend is implemented using **Flask** and Express, responsible for handling API requests, storing annotations, and communicating with the AI models.

File and Directory Structure

- **`/frontend`** - React-based user interface.
- **`/backend`** - Flask (or Express) backend handling annotation data and API endpoints.
- **`/ritm_interactive_segmentation`** - RITM demo and supporting scripts for mask-to-JSON conversion and related utilities.
- **`/XMem2-cpu-web`** - XMem inference engine for video object segmentation and mask propagation.
- **`/src/JPEGImages`** - Input image directory used for annotation.
- **`/src/mask-rit`** - Directory where RITM-generated masks are saved.
- **`/src/json`** - Contains final JSON annotation files exported from the tool.
- **`/src/Annotations`** - Stores masks from manual annotations or other external tools.

Important Notes and Best Practices

- **Model Switching Delay**:
  When switching between RITM and XMem workflows, please allow a few seconds for the APIs to initialize. This delay occurs because the system dynamically switches between the main Express thread (used by XMem) and the Flask-based RITM API.

- **RITM Click Behavior**:
  If you place two points that correspond to the same object but are very far apart in the RITM interface, the model may interpret them as separate instances. This can be corrected manually in polygon editing mode after the initial segmentation.

- **Virtual Environment Setup**:
  The `server.js` file expects two separate Python virtual environments:
  One named `venv` for XMem, Another named `eenv` for RITM This separation is necessary due to conflicting dependencies between the two models. The backend handles switching between these environments automatically, but **you must ensure that `requirements.txt` files in both submodules are installed correctly**.

- **Session Persistence and Export**:
  All masks and annotations created during a session are persisted. Users can export both the segmentation masks and the corresponding JSON polygon annotations at any time.