***Detail Desing***

***Architecture***

The architecture chosen for our project is MVC (Model-View-Controller).

MVC in Our Project:

*Model:*

* Manages data related to puzzle pieces, target image analysis, and processing results.
* Utilizes OpenCV and Panoptic Segmentation for image processing and SIFT for feature matching.
* Handles storage and retrieval of analyzed puzzle piece data.

*View:*

* User-friendly graphical interface designed using a suitable Python GUI library (e.g., Tkinter or PyQt).
* Displays instructions, visualizations, and results.
* Accepts user inputs such as uploading images and selecting options.

*Controller :*

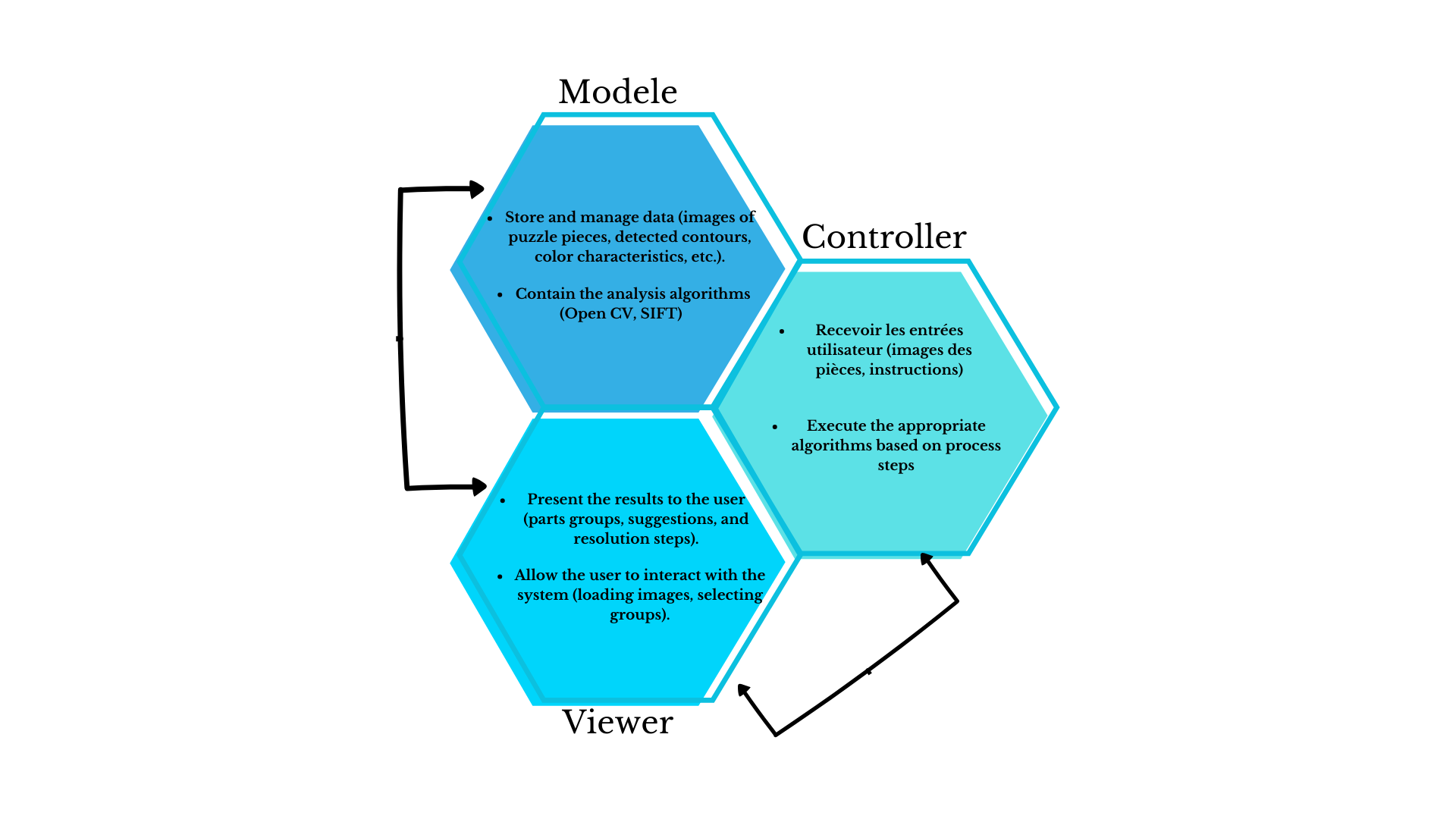
* Acts as a bridge between the Model and the View.
* Processes user inputs, triggers image analysis functions, and updates the View with results.

***Data Storage :***

Data related to the puzzle-solving process is managed as follows:

1. Uploaded Puzzle Piece Images: Temporarily stored for processing.
2. Analyzed Data: Includes contours, shapes, and color classifications stored as JSON files.
3. Target Image Data: Segmented objects and their associated color groups stored for matching.

***Graphic Description :***

****

**Key Functionalities :**

1. Outline Creation

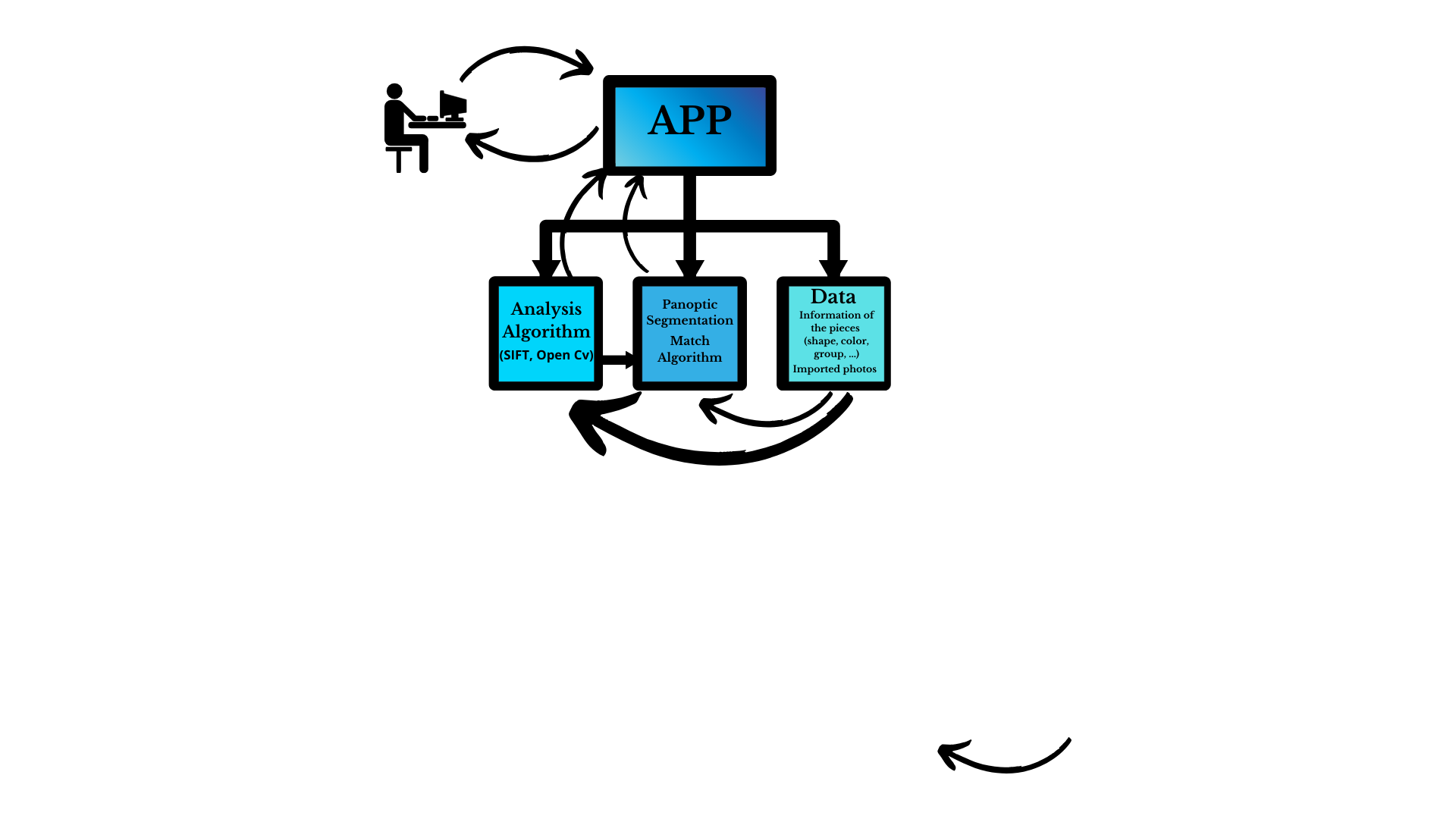
* Purpose : Assist users in starting the puzzle by separating edge pieces from inner pieces.
* Implementation :
  + Users upload images of the puzzle pieces.
  + The system analyzes these images using OpenCV functions, such as:
    - cv2.findContours for contour detection.
    - Geometric filtering to identify straight edges (indicative of border pieces).
  + Outputs two groups: edge pieces and inner pieces.

*2. Color-Based Classification*

* Purpose : Group puzzle pieces by color to simplify assembly.
* Implementation :
  + Analyze the target image to segment it into distinct objects based on color using image segmentation techniques.
    - Store color data for each segment.
  + Compare puzzle piece images against the target image using SIFT for feature matching.
    - Assign each piece to the most similar color group.

*3. Shape Classification*

* Purpose : Further classify pieces within color groups based on shape.
* Implementation :
  + Analyze contours of each piece to determine the number of curved edges (one, two, three, or four).
  + Group pieces accordingly to facilitate logical assembly.

***Interface Design***

***Algorithmic Details***

*Image Processing Algorithms*

* Contour Detection :
  + Function : cv2.findContours.
  + Extracts edges and shapes from puzzle pieces.
* Geometric Filtering :
  + Identifies straight edges for border pieces.
* Color Analysis :
* Uses panoptic segmentation for object segmentation in imageFeature Matching:
  + SIFT (Scale-Invariant Feature Transform):
    - Matches puzzle pieces to target image segments under scale and rotation variations.

*Feature Matching Algorithm :*

* Compare each puzzle piece image to segments in the target image.
* Assign matching pieces to the corresponding object or color group.

*Classification of Shapes :*

* Analyze the contours for geometric features.
* Categorize based on the number of curved edges (e.g., corners vs. edges).

***User Interface Design***

*Inputs*

* Upload images of puzzle pieces.
* Upload target image for analysis.
* Select desired classifications (e.g., color, shape).

*Outputs*

* *Visual grouping of puzzle pieces:* 
  + *Edge vs. inner pieces.*
  + Color-based groups.
  + Shape-based classifications.
* Step-by-step guidance for assembly.

***Tools and Libraries***

* *OpenCV*: For image analysis and processing.
* *SIFT* : For feature matching.
* *Python GUI Library*: Tkinter or PyQt for the user interface.

***Performance Considerations***

* Optimize image processing to handle large datasets efficiently.
* Use multi-threading or parallel processing for real-time analysis.
* Ensure the interface remains responsive during processing.

***Future Enhancements***

* Include a "hints" feature to suggest next steps for puzzle assembly.
* Extend support for 3D puzzles using depth analysis.
* Integrate machine learning to predict challenging areas of assembly based on user progress.