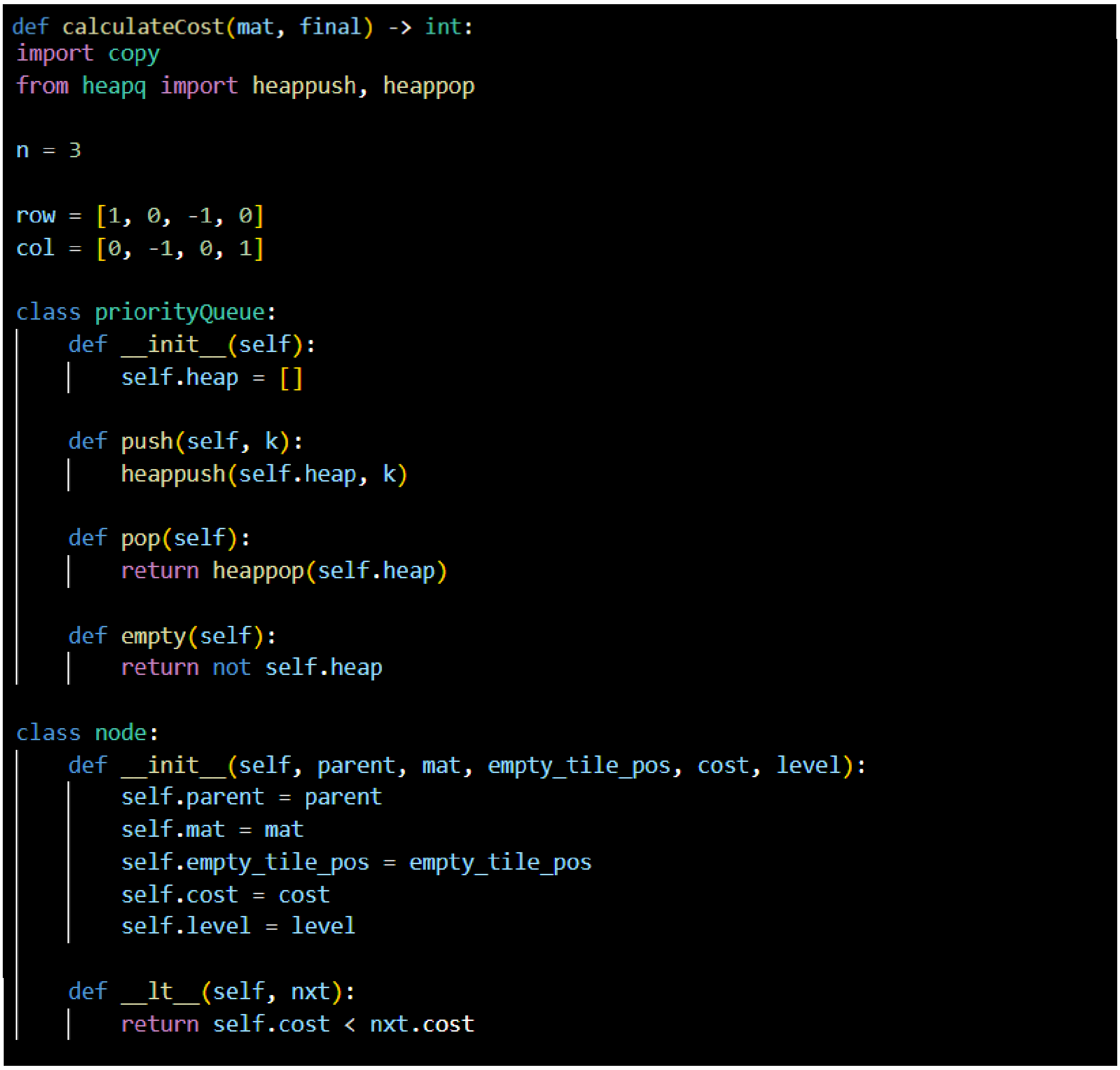
AI APPLICATIONS

HU22CSEN0100999

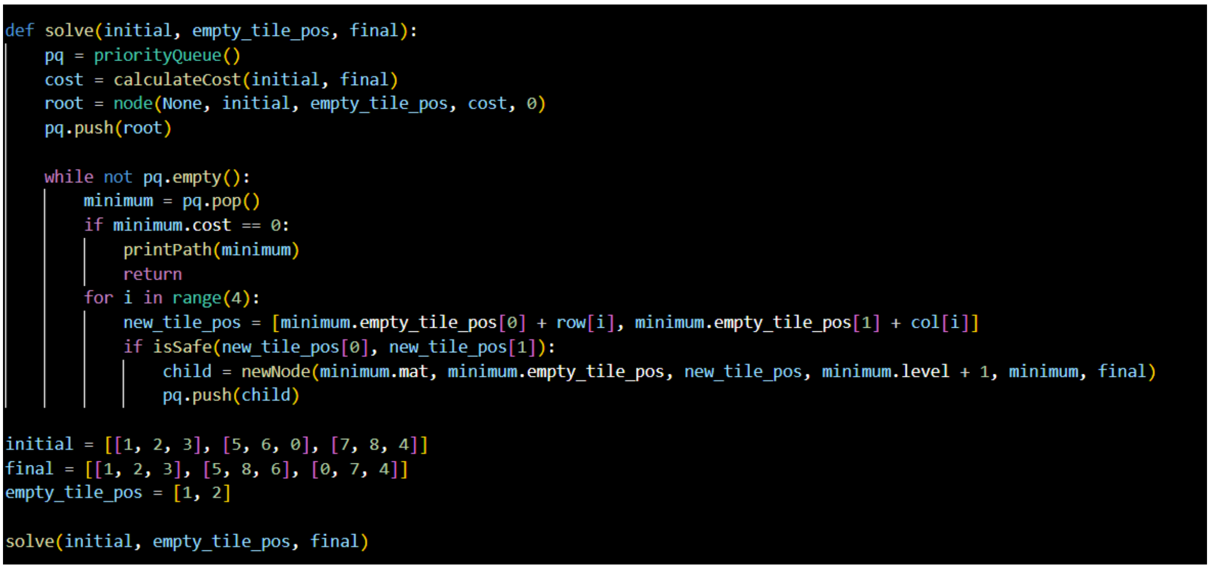
Eshwar Deshmukh Chavan

Upload screenshots of code and execution of 8-puzzle problem, Wumpus World, Vacuum cleaner, Sudoku and Crossword puzzle.

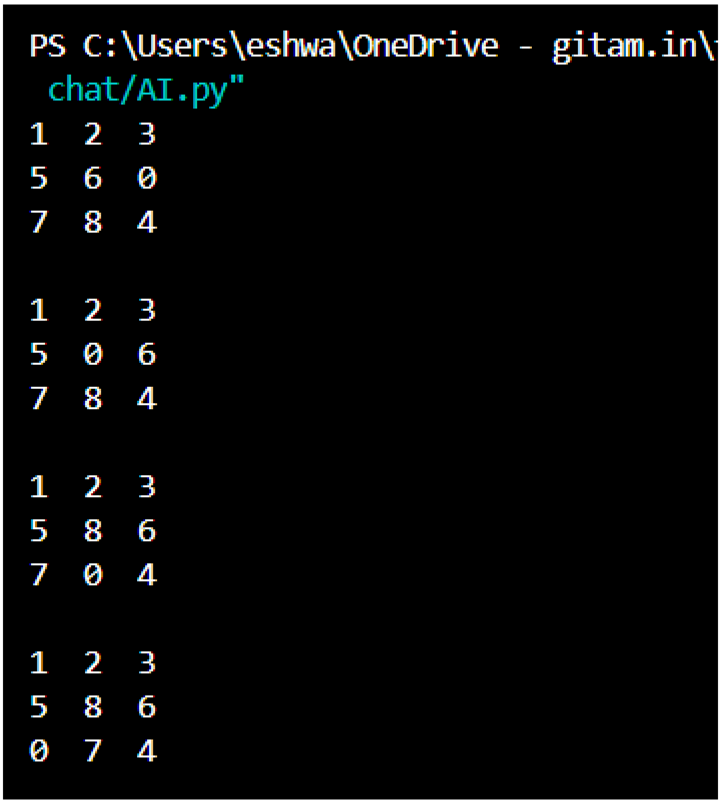
8-PUZZLE PROBLEM



```
def calculateCost(mat, final) -> int:
import copy
from heapq import heappush, heappop

n = 3

row = [1, 0, -1, 0]
col = [0, -1, 0, 1]

class priorityQueue:
    def __init__(self):
        self.heap = []

    def push(self, k):
        heappush(self.heap, k)

    def pop(self):
        return heappop(self.heap)

    def empty(self):
        return not self.heap

class node:
    def __init__(self, parent, mat, empty_tile_pos, cost, level):
        self.parent = parent
        self.mat = mat
        self.empty_tile_pos = empty_tile_pos
        self.cost = cost
        self.level = level

    def __lt__(self, nxt):
        return self.cost < nxt.cost
```

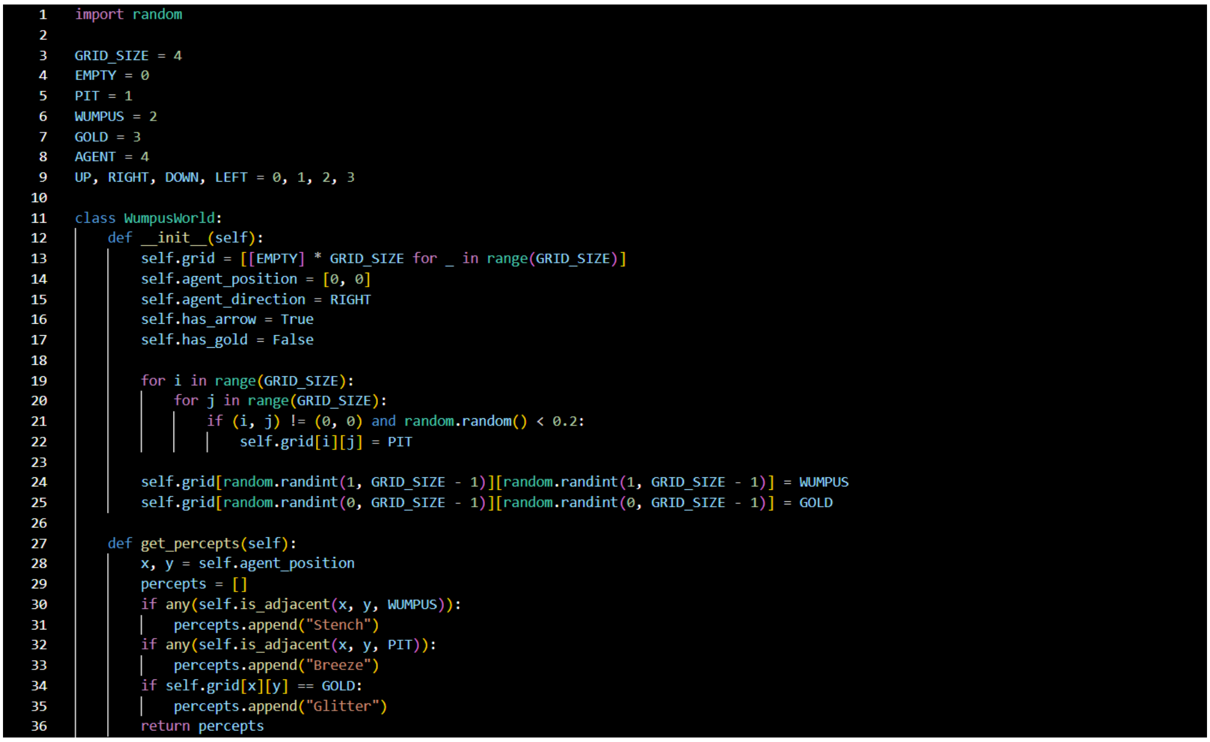


```
def solve(initial, empty_tile_pos, final):
    pq = priorityQueue()
    cost = calculateCost(initial, final)
    root = node(None, initial, empty_tile_pos, cost, 0)
    pq.push(root)

    while not pq.empty():
        minimum = pq.pop()
        if minimum.cost == 0:
            printPath(minimum)
            return
        for i in range(4):
            new_tile_pos = [minimum.empty_tile_pos[0] + row[i], minimum.empty_tile_pos[1] + col[i]]
            if isSafe(new_tile_pos[0], new_tile_pos[1]):
                child = newNode(minimum.mat, minimum.empty_tile_pos, new_tile_pos, minimum.level + 1, minimum, final)
                pq.push(child)

initial = [[1, 2, 3], [5, 6, 0], [7, 8, 4]]
final = [[1, 2, 3], [5, 8, 6], [0, 7, 4]]
empty_tile_pos = [1, 2]

solve(initial, empty_tile_pos, final)
```

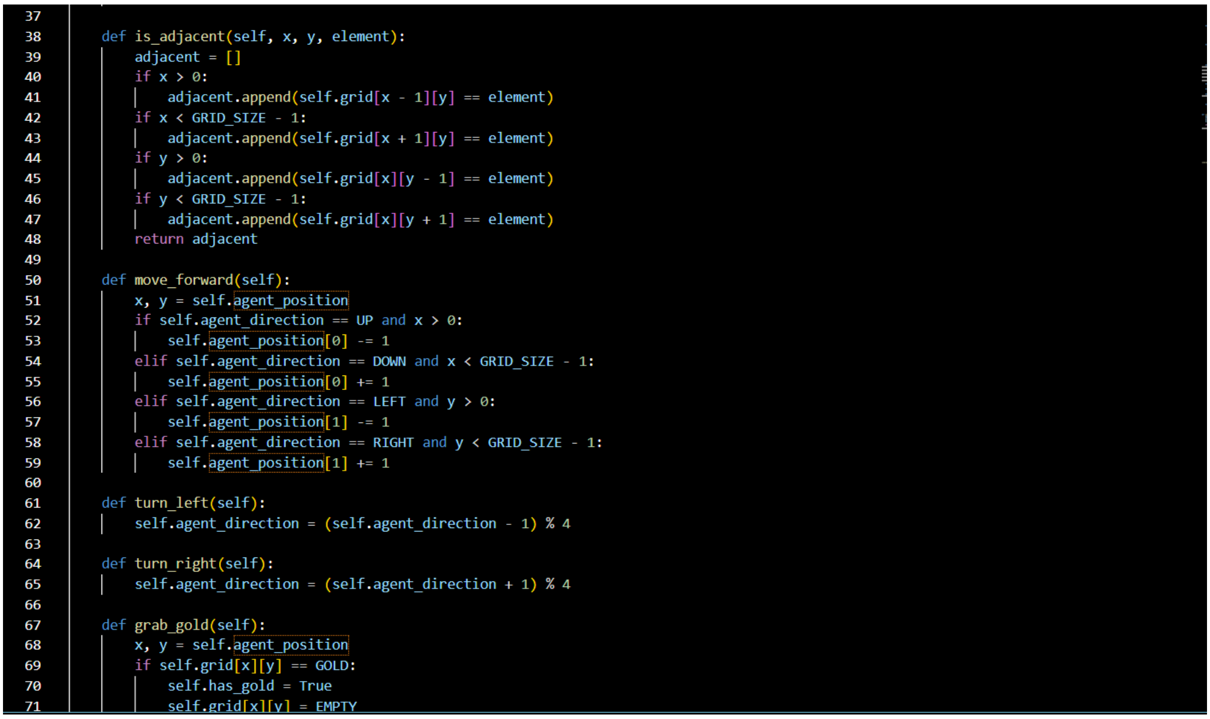


```
PS C:\Users\eshwa\OneDrive - gitam.in\
 chat/AI.py"
1  2  3
5  6  0
7  8  4

1  2  3
5  0  6
7  8  4

1  2  3
5  8  6
7  0  4

1  2  3
5  8  6
0  7  4
```

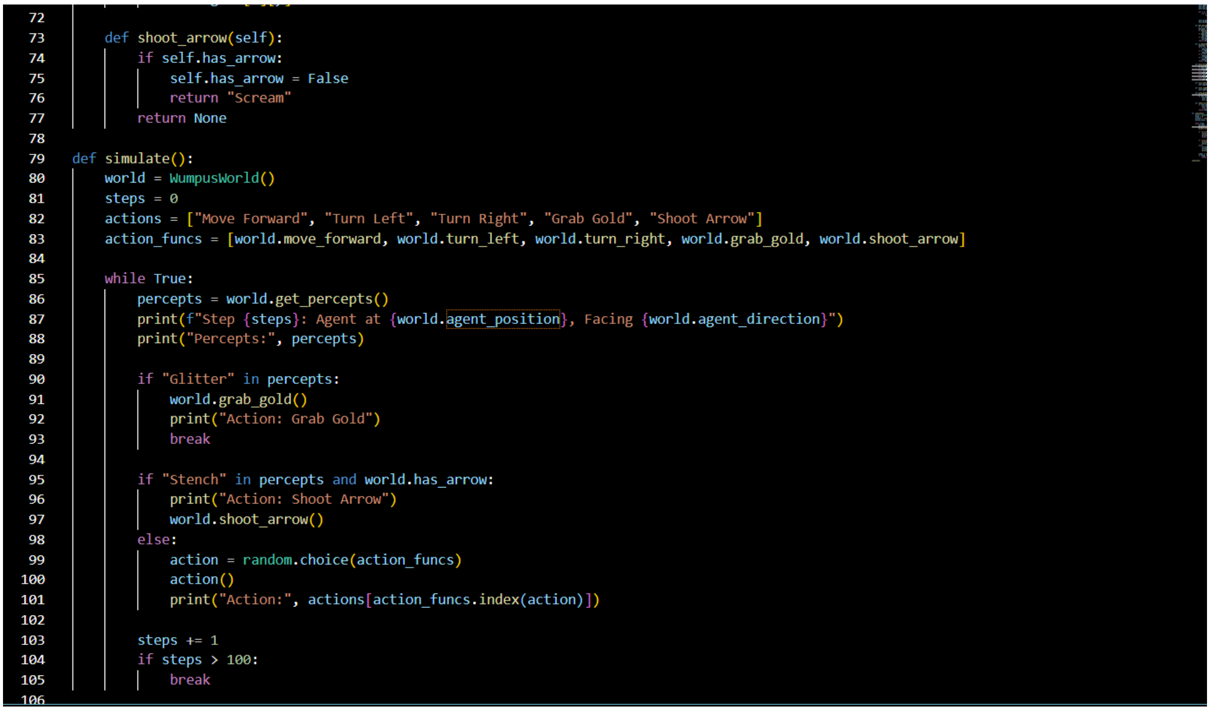
WUMPUS WORLD PROBLEM:



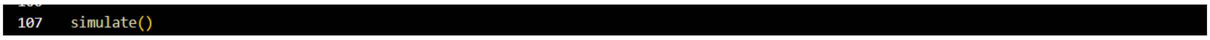
```python
import random

GRID_SIZE = 4
EMPTY = 0
PIT = 1
WUMPUS = 2
GOLD = 3
AGENT = 4
UP, RIGHT, DOWN, LEFT = 0, 1, 2, 3

class WumpusWorld:
    def __init__(self):
        self.grid = [[EMPTY] * GRID_SIZE for _ in range(GRID_SIZE)]
        self.agent_position = [0, 0]
        self.agent_direction = RIGHT
        self.has_arrow = True
        self.has_gold = False

        for i in range(GRID_SIZE):
            for j in range(GRID_SIZE):
                if (i, j) != (0, 0) and random.random() < 0.2:
                    self.grid[i][j] = PIT

        self.grid[random.randint(1, GRID_SIZE - 1)][random.randint(1, GRID_SIZE - 1)] = WUMPUS
        self.grid[random.randint(0, GRID_SIZE - 1)][random.randint(0, GRID_SIZE - 1)] = GOLD

    def get_percepts(self):
        x, y = self.agent_position
        percepts = []
        if any(self.is_adjacent(x, y, WUMPUS)):
            percepts.append("Stench")
        if any(self.is_adjacent(x, y, PIT)):
            percepts.append("Breeze")
        if self.grid[x][y] == GOLD:
            percepts.append("Glitter")
        return percepts
```



```
    def is_adjacent(self, x, y, element):
        adjacent = []
        if x > 0:
            adjacent.append(self.grid[x - 1][y] == element)
        if x < GRID_SIZE - 1:
            adjacent.append(self.grid[x + 1][y] == element)
        if y > 0:
            adjacent.append(self.grid[x][y - 1] == element)
        if y < GRID_SIZE - 1:
            adjacent.append(self.grid[x][y + 1] == element)
        return adjacent

    def move_forward(self):
        x, y = self.agent_position
        if self.agent_direction == UP and x > 0:
            self.agent_position[0] -= 1
        elif self.agent_direction == DOWN and x < GRID_SIZE - 1:
            self.agent_position[0] += 1
        elif self.agent_direction == LEFT and y > 0:
            self.agent_position[1] -= 1
        elif self.agent_direction == RIGHT and y < GRID_SIZE - 1:
            self.agent_position[1] += 1

    def turn_left(self):
        self.agent_direction = (self.agent_direction - 1) % 4

    def turn_right(self):
        self.agent_direction = (self.agent_direction + 1) % 4

    def grab_gold(self):
        x, y = self.agent_position
        if self.grid[x][y] == GOLD:
            self.has_gold = True
            self.grid[x][y] = EMPTY
```

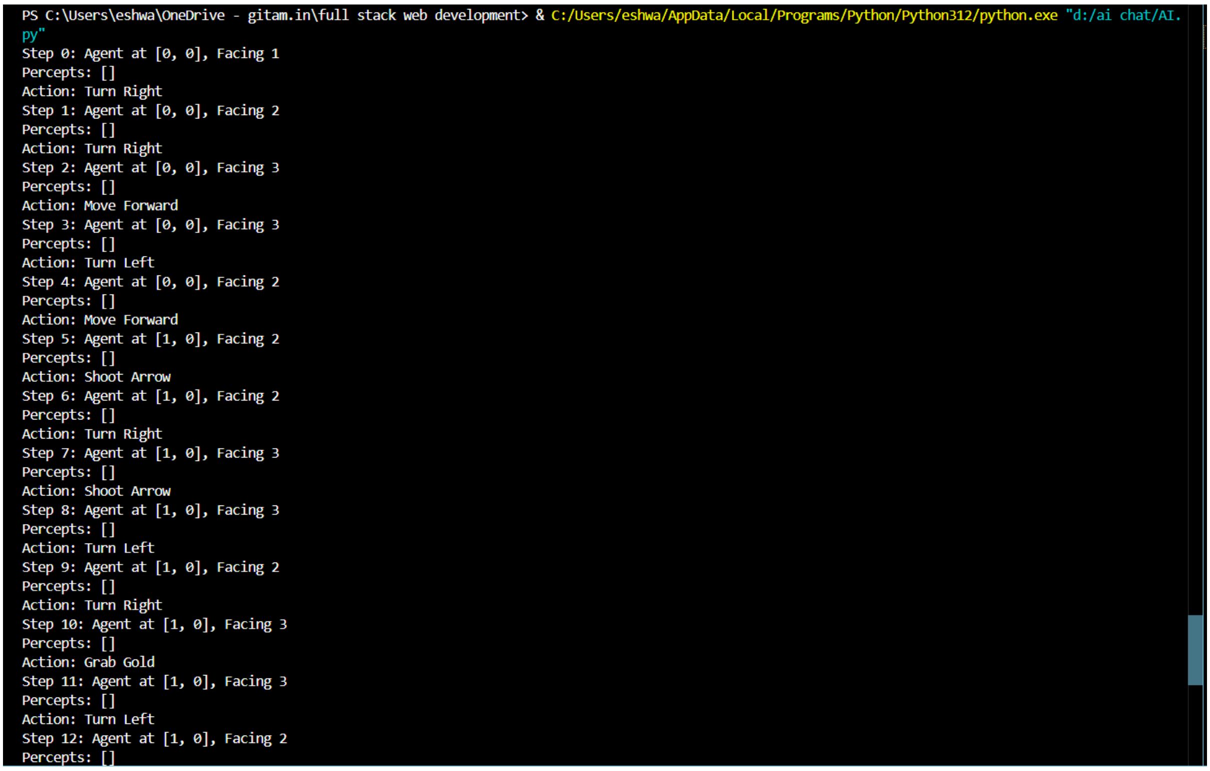


```

    def shoot_arrow(self):
        if self.has_arrow:
            self.has_arrow = False
            return "Scream"
        return None

def simulate():
    world = WumpusWorld()
    steps = 0
    actions = ["Move Forward", "Turn Left", "Turn Right", "Grab Gold", "Shoot Arrow"]
    action_funcs = [world.move_forward, world.turn_left, world.turn_right, world.grab_gold, world.shoot_arrow]

    while True:
        percepts = world.get_percepts()
        print(f"Step {steps}: Agent at {world.agent_position}, Facing {world.agent_direction}")
        print("Percepts:", percepts)

        if "Glitter" in percepts:
            world.grab_gold()
            print("Action: Grab Gold")
            break

        if "Stench" in percepts and world.has_arrow:
            print("Action: Shoot Arrow")
            world.shoot_arrow()
        else:
            action = random.choice(action_funcs)
            action()
            print("Action:", actions[action_funcs.index(action)])

        steps += 1
        if steps > 100:
            break
```

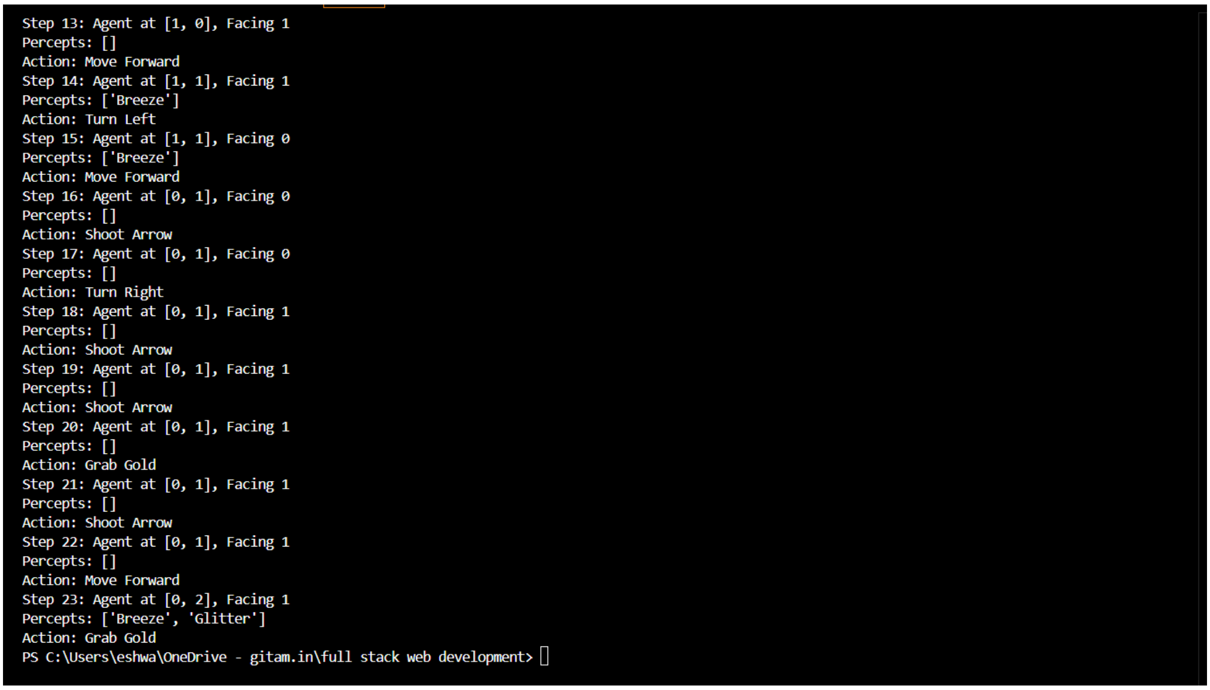


```
simulate()
```

Output:

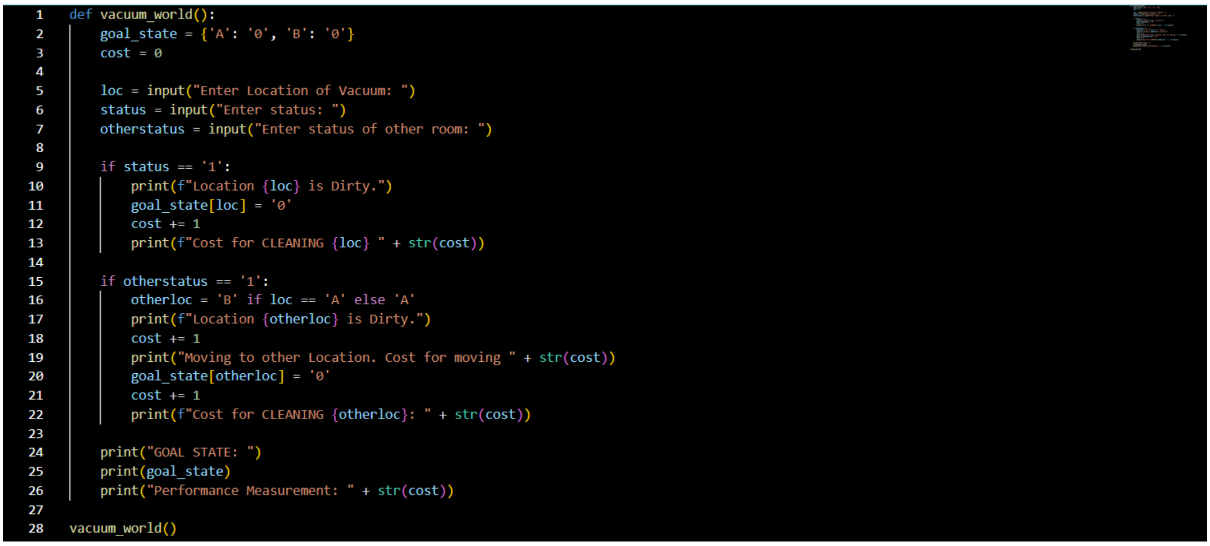


```
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development> & C:/Users/eshwa/AppData/Local/Programs/Python/Python312/python.exe "d:/ai chat/AI.
py"
Step 0: Agent at [0, 0], Facing 1
Percepts: []
Action: Turn Right
Step 1: Agent at [0, 0], Facing 2
Percepts: []
Action: Turn Right
Step 2: Agent at [0, 0], Facing 3
Percepts: []
Action: Move Forward
Step 3: Agent at [0, 0], Facing 3
Percepts: []
Action: Turn Left
Step 4: Agent at [0, 0], Facing 2
Percepts: []
Action: Move Forward
Step 5: Agent at [1, 0], Facing 2
Percepts: []
Action: Shoot Arrow
Step 6: Agent at [1, 0], Facing 2
Percepts: []
Action: Turn Right
Step 7: Agent at [1, 0], Facing 3
Percepts: []
Action: Shoot Arrow
Step 8: Agent at [1, 0], Facing 3
Percepts: []
Action: Turn Left
Step 9: Agent at [1, 0], Facing 2
Percepts: []
Action: Turn Right
Step 10: Agent at [1, 0], Facing 3
Percepts: []
Action: Grab Gold
Step 11: Agent at [1, 0], Facing 3
Percepts: []
Action: Turn Left
Step 12: Agent at [1, 0], Facing 2
Percepts: []
```



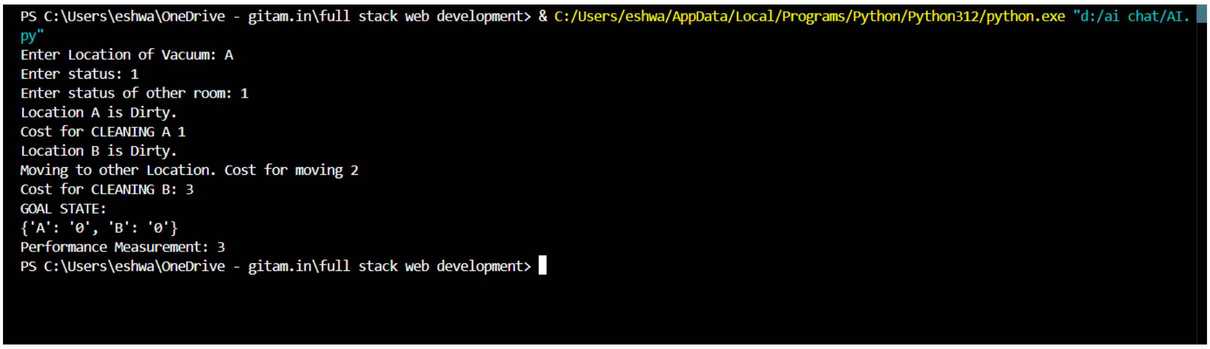
```
Step 13: Agent at [1, 0], Facing 1
Percepts: []
Action: Move Forward
Step 14: Agent at [1, 1], Facing 1
Percepts: ['Breeze']
Action: Turn Left
Step 15: Agent at [1, 1], Facing 0
Percepts: ['Breeze']
Action: Move Forward
Step 16: Agent at [0, 1], Facing 0
Percepts: []
Action: Shoot Arrow
Step 17: Agent at [0, 1], Facing 0
Percepts: []
Action: Turn Right
Step 18: Agent at [0, 1], Facing 1
Percepts: []
Action: Shoot Arrow
Step 19: Agent at [0, 1], Facing 1
Percepts: []
Action: Shoot Arrow
Step 20: Agent at [0, 1], Facing 1
Percepts: []
Action: Grab Gold
Step 21: Agent at [0, 1], Facing 1
Percepts: []
Action: Shoot Arrow
Step 22: Agent at [0, 1], Facing 1
Percepts: []
Action: Move Forward
Step 23: Agent at [0, 2], Facing 1
Percepts: ['Breeze', 'Glitter']
Action: Grab Gold
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development> 
```

VACUUM CLEANER PROBLEM:



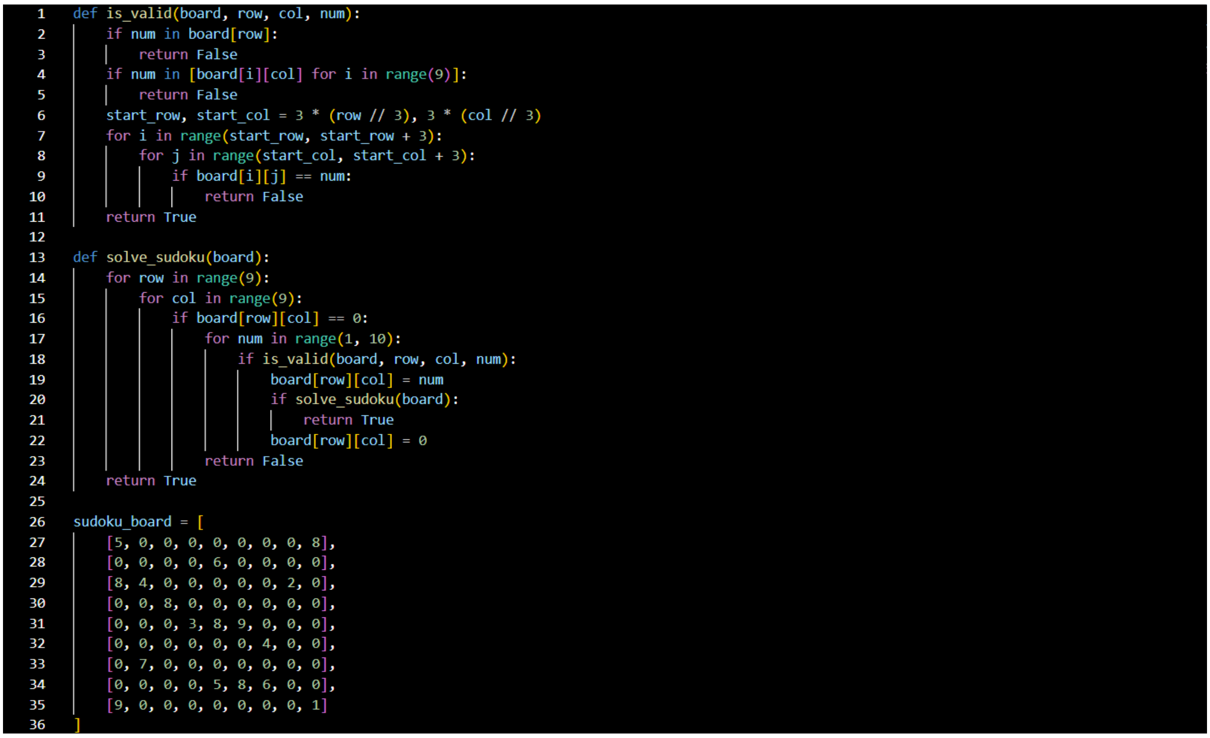
```python
def vacuum_world():
    goal_state = {'A': '0', 'B': '0'}
    cost = 0

    loc = input("Enter Location of Vacuum: ")
    status = input("Enter status: ")
    otherstatus = input("Enter status of other room: ")

    if status == '1':
        print(f"Location {loc} is Dirty.")
        goal_state[loc] = '0'
        cost += 1
        print(f"Cost for CLEANING {loc} " + str(cost))

    if otherstatus == '1':
        otherloc = 'B' if loc == 'A' else 'A'
        print(f"Location {otherloc} is Dirty.")
        cost += 1
        print("Moving to other Location. Cost for moving " + str(cost))
        goal_state[otherloc] = '0'
        cost += 1
        print(f"Cost for CLEANING {otherloc}: " + str(cost))

    print("GOAL STATE: ")
    print(goal_state)
    print("Performance Measurement: " + str(cost))

vacuum_world()
```

OUTPUT:

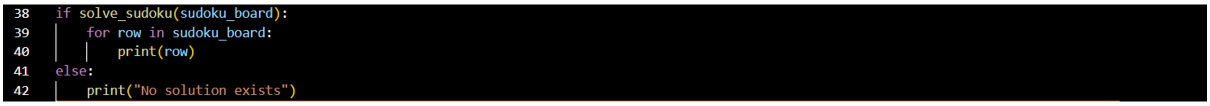


```
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development> & C:/Users/eshwa/AppData/Local/Programs/Python/Python312/python.exe "d:/ai chat/AI.
py"
Enter Location of Vacuum: A
Enter status: 1
Enter status of other room: 1
Location A is Dirty.
Cost for CLEANING A 1
Location B is Dirty.
Moving to other Location. Cost for moving 2
Cost for CLEANING B: 3
GOAL STATE:
{'A': '0', 'B': '0'}
Performance Measurement: 3
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development>
```

SUDOKU CODE:

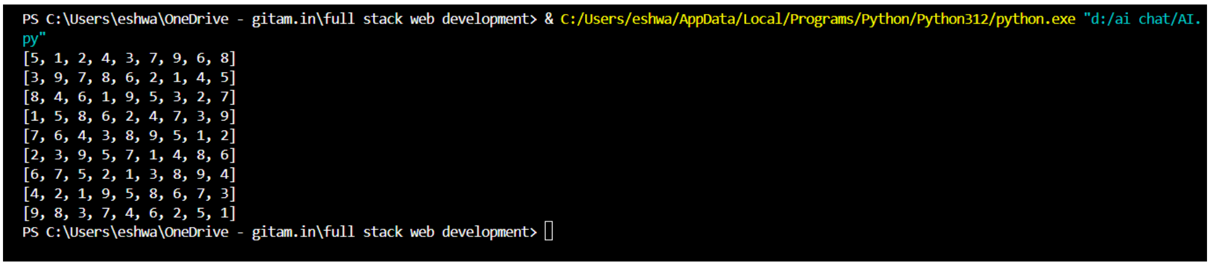


```python
def is_valid(board, row, col, num):
    if num in board[row]:
        return False
    if num in [board[i][col] for i in range(9)]:
        return False
    start_row, start_col = 3 * (row // 3), 3 * (col // 3)
    for i in range(start_row, start_row + 3):
        for j in range(start_col, start_col + 3):
            if board[i][j] == num:
                return False
    return True

def solve_sudoku(board):
    for row in range(9):
        for col in range(9):
            if board[row][col] == 0:
                for num in range(1, 10):
                    if is_valid(board, row, col, num):
                        board[row][col] = num
                        if solve_sudoku(board):
                            return True
                        board[row][col] = 0
                return False
    return True

sudoku_board = [
    [5, 0, 0, 0, 0, 0, 0, 0, 8],
    [0, 0, 0, 0, 6, 0, 0, 0, 0],
    [8, 4, 0, 0, 0, 0, 0, 2, 0],
    [0, 0, 8, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 3, 8, 9, 0, 0, 0],
    [0, 0, 0, 0, 0, 0, 4, 0, 0],
    [0, 7, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 0, 0, 5, 8, 6, 0, 0],
    [9, 0, 0, 0, 0, 0, 0, 0, 1]
]
```



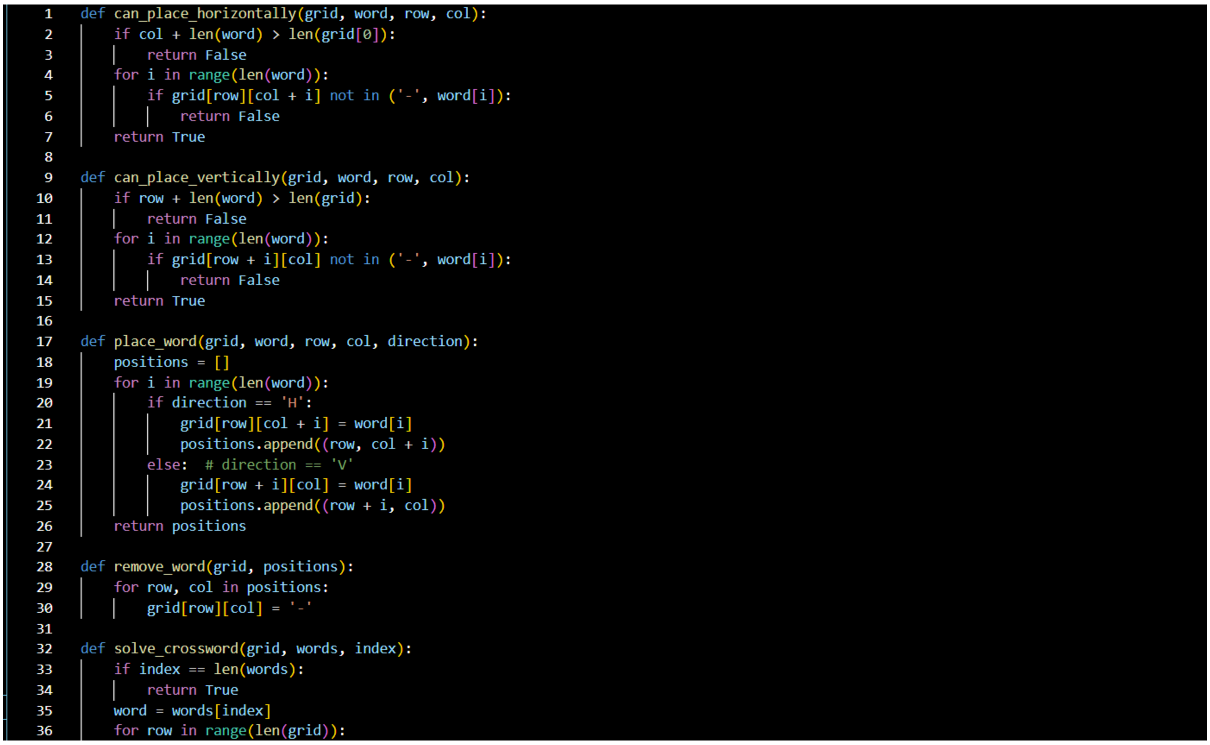
```python
if solve_sudoku(sudoku_board):
    for row in sudoku_board:
        print(row)
else:
    print("No solution exists")
```

OUTPUT:

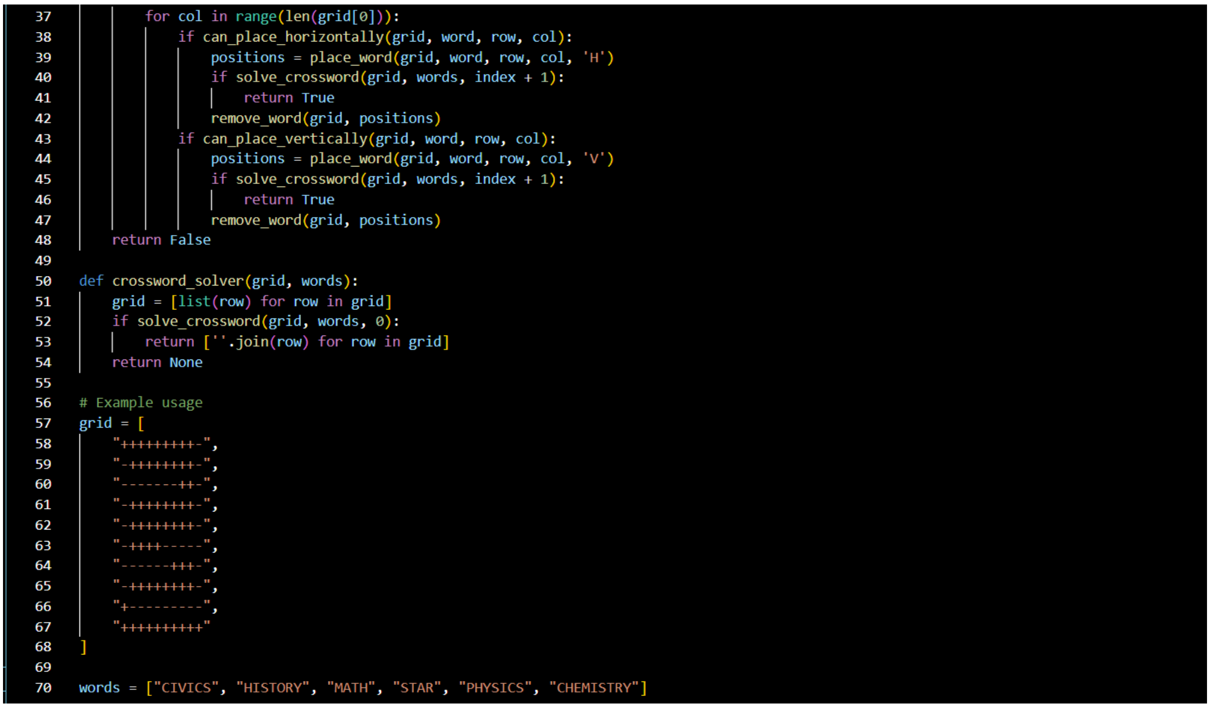


```
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development> & C:/Users/eshwa/AppData/Local/Programs/Python/Python312/python.exe "d:/ai chat/AI.
py"
[5, 1, 2, 4, 3, 7, 9, 6, 8]
[3, 9, 7, 8, 6, 2, 1, 4, 5]
[8, 4, 6, 1, 9, 5, 3, 2, 7]
[1, 5, 8, 6, 2, 4, 7, 3, 9]
[7, 6, 4, 3, 8, 9, 5, 1, 2]
[2, 3, 9, 5, 7, 1, 4, 8, 6]
[6, 7, 5, 2, 1, 3, 8, 9, 4]
[4, 2, 1, 9, 5, 8, 6, 7, 3]
[9, 8, 3, 7, 4, 6, 2, 5, 1]
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development> 
```

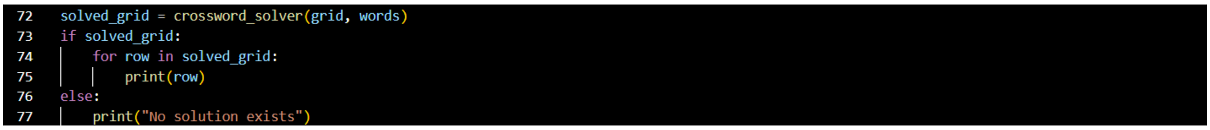
CROSSWORD PUZZLE:



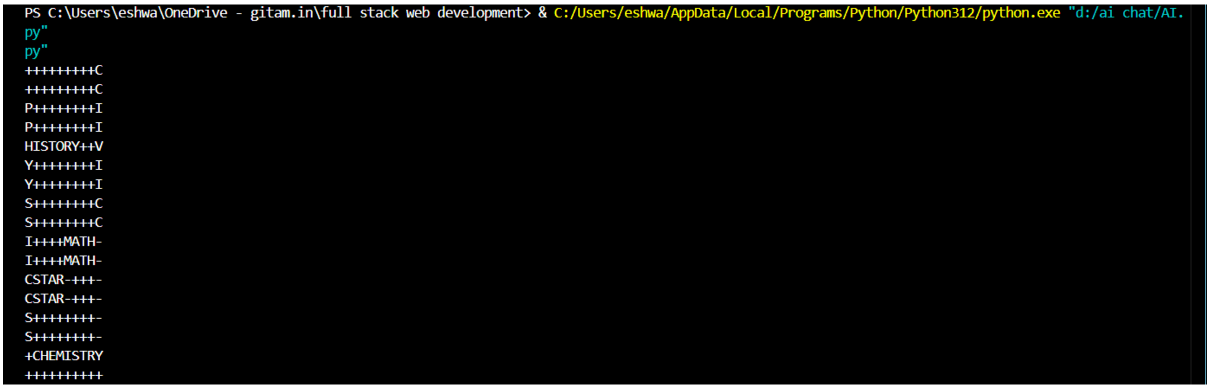
```python
def can_place_horizontally(grid, word, row, col):
    if col + len(word) > len(grid[0]):
        return False
    for i in range(len(word)):
        if grid[row][col + i] not in ('-', word[i]):
            return False
    return True

def can_place_vertically(grid, word, row, col):
    if row + len(word) > len(grid):
        return False
    for i in range(len(word)):
        if grid[row + i][col] not in ('-', word[i]):
            return False
    return True

def place_word(grid, word, row, col, direction):
    positions = []
    for i in range(len(word)):
        if direction == 'H':
            grid[row][col + i] = word[i]
            positions.append((row, col + i))
        else:  # direction == 'V'
            grid[row + i][col] = word[i]
            positions.append((row + i, col))
    return positions

def remove_word(grid, positions):
    for row, col in positions:
        grid[row][col] = '-'

def solve_crossword(grid, words, index):
    if index == len(words):
        return True
    word = words[index]
    for row in range(len(grid)):
```



```python
        for col in range(len(grid[0])):
            if can_place_horizontally(grid, word, row, col):
                positions = place_word(grid, word, row, col, 'H')
                if solve_crossword(grid, words, index + 1):
                    return True
                remove_word(grid, positions)
            if can_place_vertically(grid, word, row, col):
                positions = place_word(grid, word, row, col, 'V')
                if solve_crossword(grid, words, index + 1):
                    return True
                remove_word(grid, positions)
    return False

def crossword_solver(grid, words):
    grid = [list(row) for row in grid]
    if solve_crossword(grid, words, 0):
        return [''.join(row) for row in grid]
    return None

# Example usage
grid = [
    "+++++++++-",
    "-++++++++-",
    "-------++-",
    "-++++++++-",
    "-++++++++-",
    "-++++-----",
    "------+++-",
    "-++++++++-",
    "+---------",
    "++++++++++"
]

words = ["CIVICS", "HISTORY", "MATH", "STAR", "PHYSICS", "CHEMISTRY"]
```



```python
solved_grid = crossword_solver(grid, words)
if solved_grid:
    for row in solved_grid:
        print(row)
else:
    print("No solution exists")
```



```
PS C:\Users\eshwa\OneDrive - gitam.in\full stack web development> & C:/Users/eshwa/AppData/Local/Programs/Python/Python312/python.exe "d:/ai chat/AI.
py"
py"
+++++++++C
+++++++++C
P++++++++I
P++++++++I
HISTORY++V
Y++++++++I
Y++++++++I
S++++++++C
S++++++++C
I++++MATH-
I++++MATH-
CSTAR-+++-
CSTAR-+++-
S++++++++-
S++++++++-
+CHEMISTRY
++++++++++
```

CONCEPTUAL INTRODUTION TO MACHINE LEARNING

Machine Learning(ML): is a branch of AI focused on developing algorithms that allow computers to learn from and make decisions based on data. Instead of relying on explicit programming. ML Models identify patterns in data to improve their performance on tasks like prediction or classifications. The Key objective is to enable machine to learn from data and make accurate decisions or predictions across various domains.

ML can be broadly classified into three types:

- Supervised Learning
- Unsupervised learning
- Semisupervised learning.

Supervised Leaning

- Use Labeled data to train models that predicts or classify outputs based on new inputs. The main types are:

→ classification: Assign input to predefined categories (eg. spam detection in mails)

→ Regression : Predicts continuous values

Common Algorithms:

- linear Regression
- Decision Tree
- Neural Network

Applications :

- Image classification
- predictive Analytics
- Sentiment Analytics

UNSUPERVISED LEARNING

- Deals with unlabeled data, aiming to uncover hidden structures or patterns. The main tasks include:

→ clustering : grouping similar data points

→ Dimensionality Reduction : simplifying data while preserving important features.

Common Algorithms:

- K-means clustering
- Principal component Analysis (PCA)

Applications:

- Anomaly Detection
- Market Basket Analysis
- Data compression

Semi Supervised Learning

- Combines a small amount of labeled data with a large amount of unlabeled data, enhancing model performance when labeling data is costly or limited. It leverages the labeled data to guide the learning process on the unlabeled data.

Common Algorithms:

- Self Training
- Graph-Based Methods.

Applications:

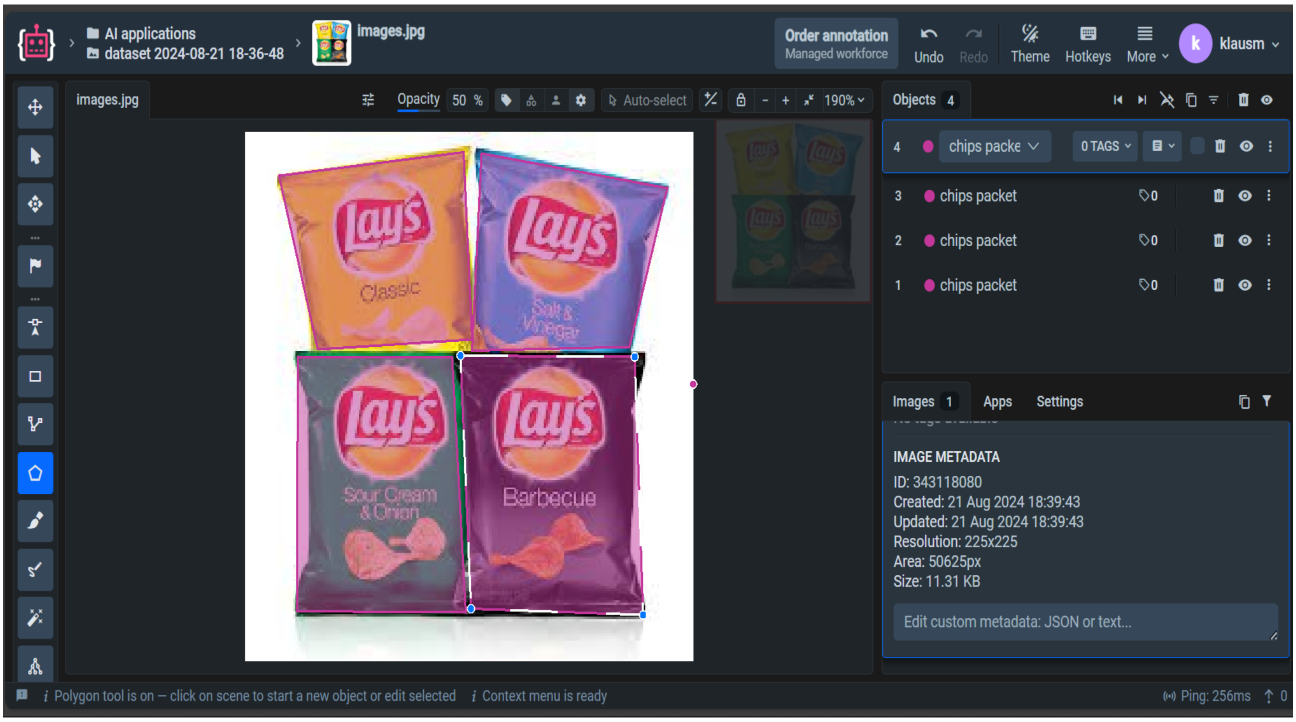
- Text classification
- Image Recognition
- Medical Image Analysis.

AI APPLICATIONS

HU22CSEN0100999

Eshwar Deshmukh Chavan

IMAGE-1

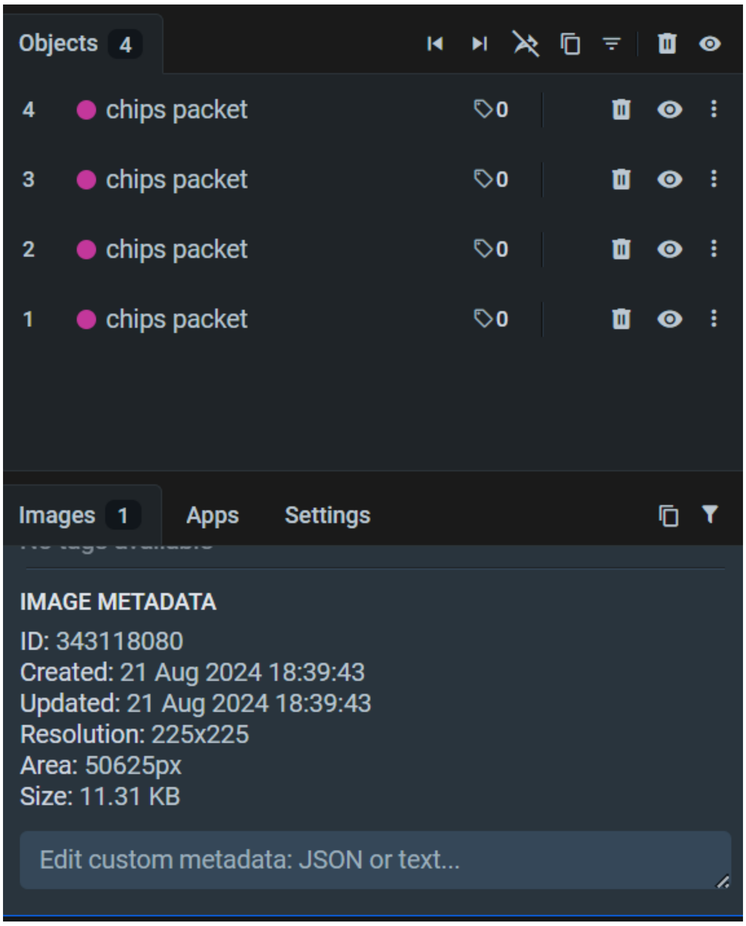


4 chips packet
3 chips packet 0
2 chips packet 0
1 chips packet 0

IMAGE METADATA
ID: 343118080
Created: 21 Aug 2024 18:39:43
Updated: 21 Aug 2024 18:39:43
Resolution: 225x225
Area: 50625px
Size: 11.31 KB

Edit custom metadata: JSON or text...

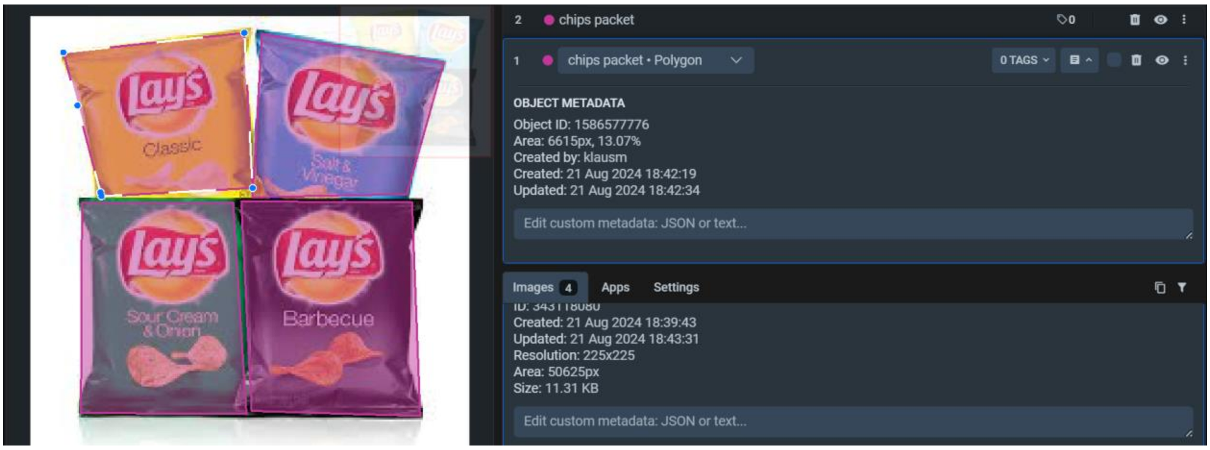
Polygon tool is on — click on scene to start a new object or edit selected
Context menu is ready
Ping: 256ms



4 chips packet 0
3 chips packet 0
2 chips packet 0
1 chips packet 0

IMAGE METADATA
ID: 343118080
Created: 21 Aug 2024 18:39:43
Updated: 21 Aug 2024 18:39:43
Resolution: 225x225
Area: 50625px
Size: 11.31 KB

Edit custom metadata: JSON or text...

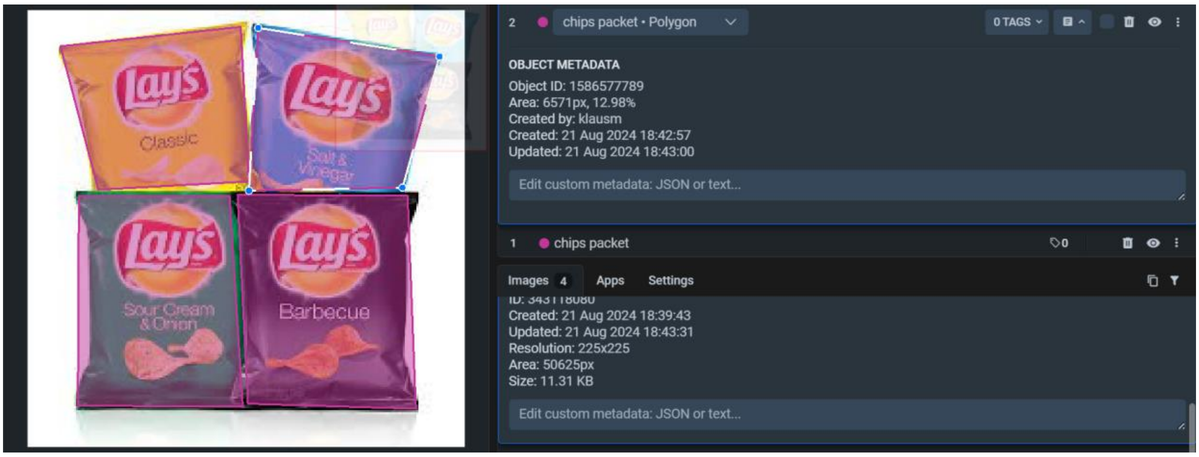


OBJECT METADATA
Object ID: 1586577776
Area: 6615px, 13.07%
Created by: klausm
Created: 21 Aug 2024 18:42:19
Updated: 21 Aug 2024 18:42:34

Edit custom metadata: JSON or text...

ID: 343118080
Created: 21 Aug 2024 18:39:43
Updated: 21 Aug 2024 18:43:31
Resolution: 225x225
Area: 50625px
Size: 11.31 KB

Edit custom metadata: JSON or text...

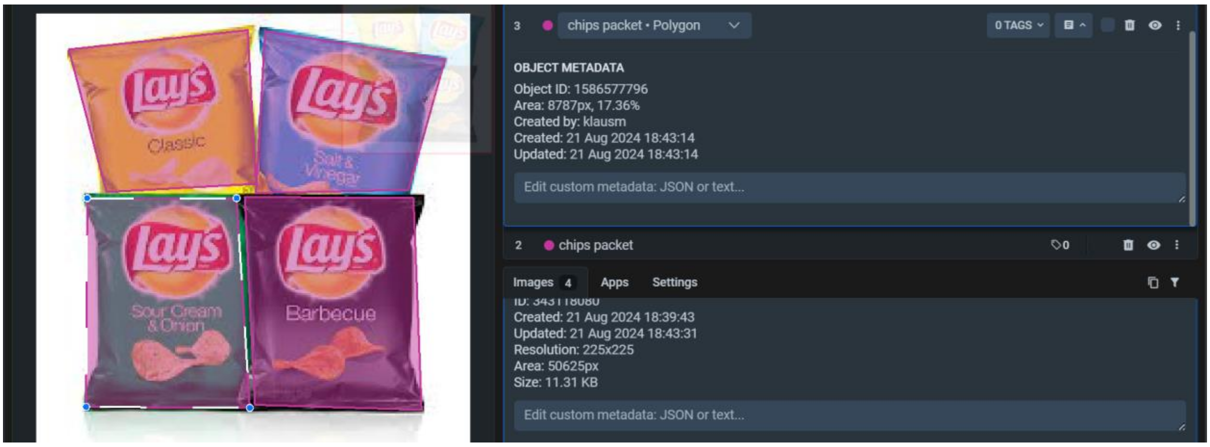


OBJECT METADATA
Object ID: 1586577789
Area: 6571px, 12.98%
Created by: klausm
Created: 21 Aug 2024 18:42:57
Updated: 21 Aug 2024 18:43:00

Edit custom metadata: JSON or text...

ID: 343118080
Created: 21 Aug 2024 18:39:43
Updated: 21 Aug 2024 18:43:31
Resolution: 225x225
Area: 50625px
Size: 11.31 KB

Edit custom metadata: JSON or text...

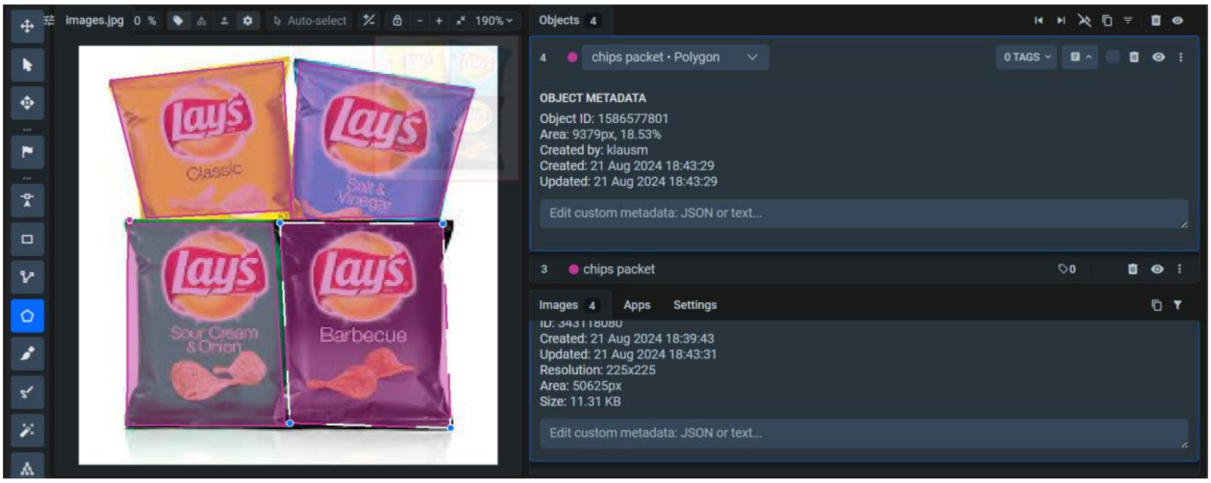


OBJECT METADATA
Object ID: 1586577796
Area: 8787px, 17.36%
Created by: klausm
Created: 21 Aug 2024 18:43:14
Updated: 21 Aug 2024 18:43:14

Edit custom metadata: JSON or text...

ID: 343118080
Created: 21 Aug 2024 18:39:43
Updated: 21 Aug 2024 18:43:31
Resolution: 225x225
Area: 50625px
Size: 11.31 KB

Edit custom metadata: JSON or text...



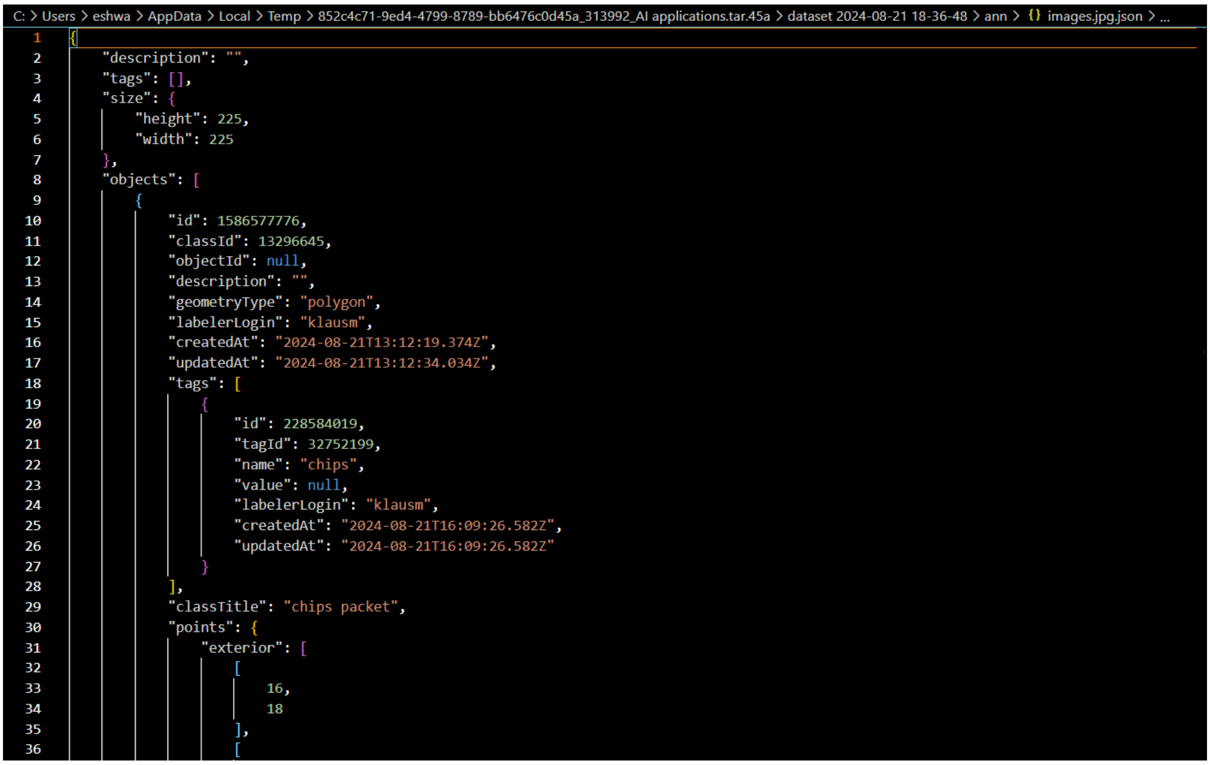
OBJECT METADATA
Object ID: 1586577801
Area: 9379px, 18.53%
Created by: klausm
Created: 21 Aug 2024 18:43:29
Updated: 21 Aug 2024 18:43:29

Edit custom metadata: JSON or text...

ID: 343118080
Created: 21 Aug 2024 18:39:43
Updated: 21 Aug 2024 18:43:31
Resolution: 225x225
Area: 50625px
Size: 11.31 KB

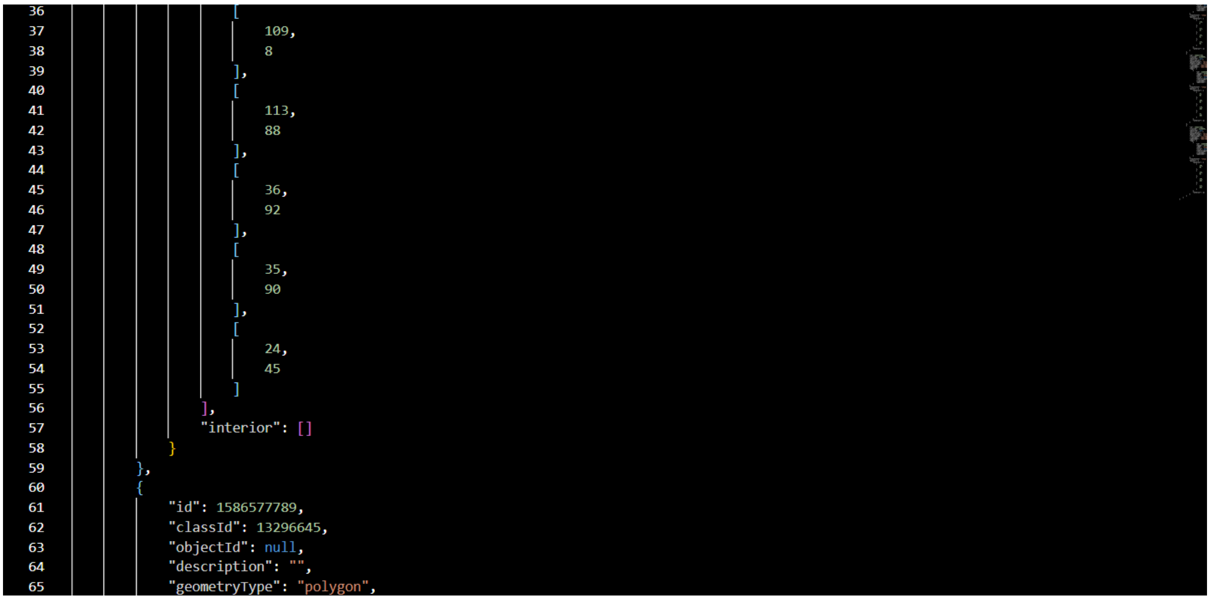
Edit custom metadata: JSON or text...

IMAGE 1 JSON:

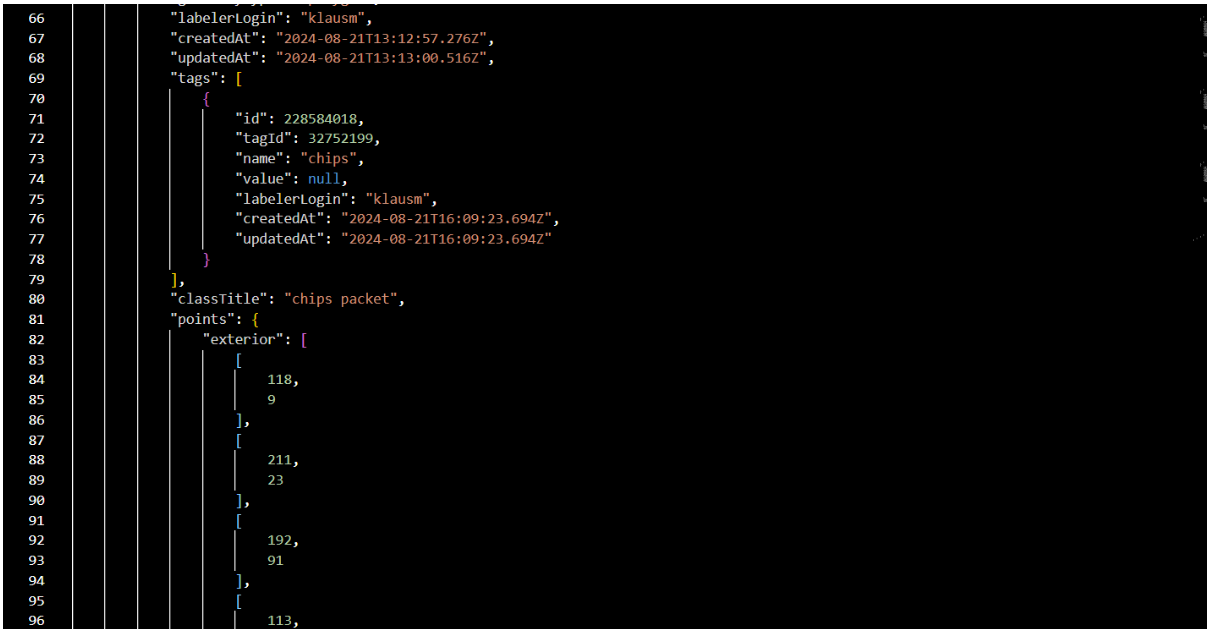


C: > Users > eshwa > AppData > Local > Temp > 852c4c71-9ed4-4799-8789-bb6476c0d45a_313992_AI applications.tar.45a > dataset 2024-08-21 18-36-48 > ann > {} images.jpg.json > ...

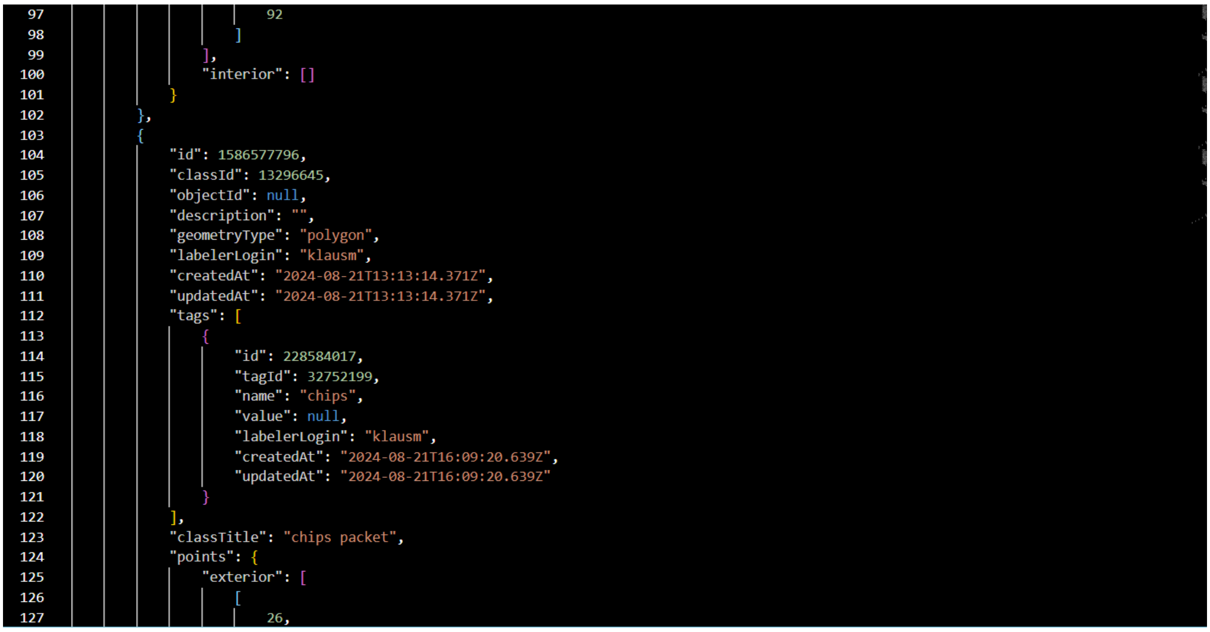
```
{
    "description": "",
    "tags": [],
    "size": {
        "height": 225,
        "width": 225
    },
    "objects": [
        {
            "id": 1586577776,
            "classId": 13296645,
            "objectId": null,
            "description": "",
            "geometryType": "polygon",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T13:12:19.374Z",
            "updatedAt": "2024-08-21T13:12:34.034Z",
            "tags": [
                {
                    "id": 228584019,
                    "tagId": 32752199,
                    "name": "chips",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:09:26.582Z",
                    "updatedAt": "2024-08-21T16:09:26.582Z"
                }
            ],
            "classTitle": "chips packet",
            "points": {
                "exterior": [
                    [
                        16,
                        18
                    ],
                    [
```



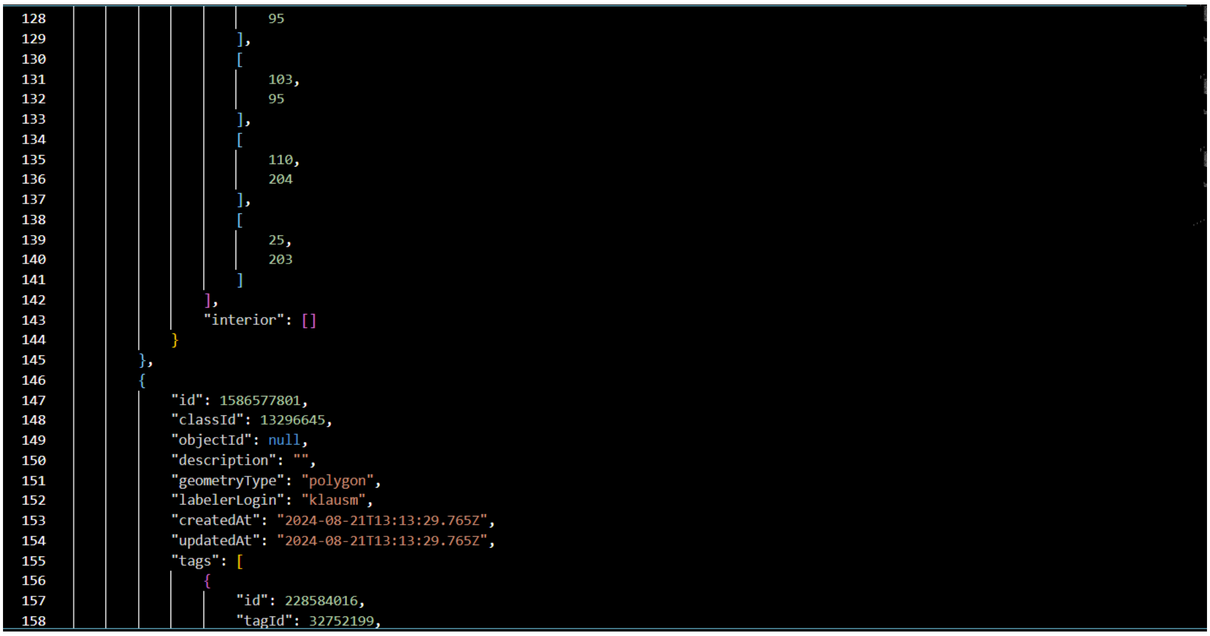
```
                    [
                        109,
                        8
                    ],
                    [
                        113,
                        88
                    ],
                    [
                        36,
                        92
                    ],
                    [
                        35,
                        90
                    ],
                    [
                        24,
                        45
                    ]
                ],
                "interior": []
            }
        },
        {
            "id": 1586577789,
            "classId": 13296645,
            "objectId": null,
            "description": "",
            "geometryType": "polygon",
```



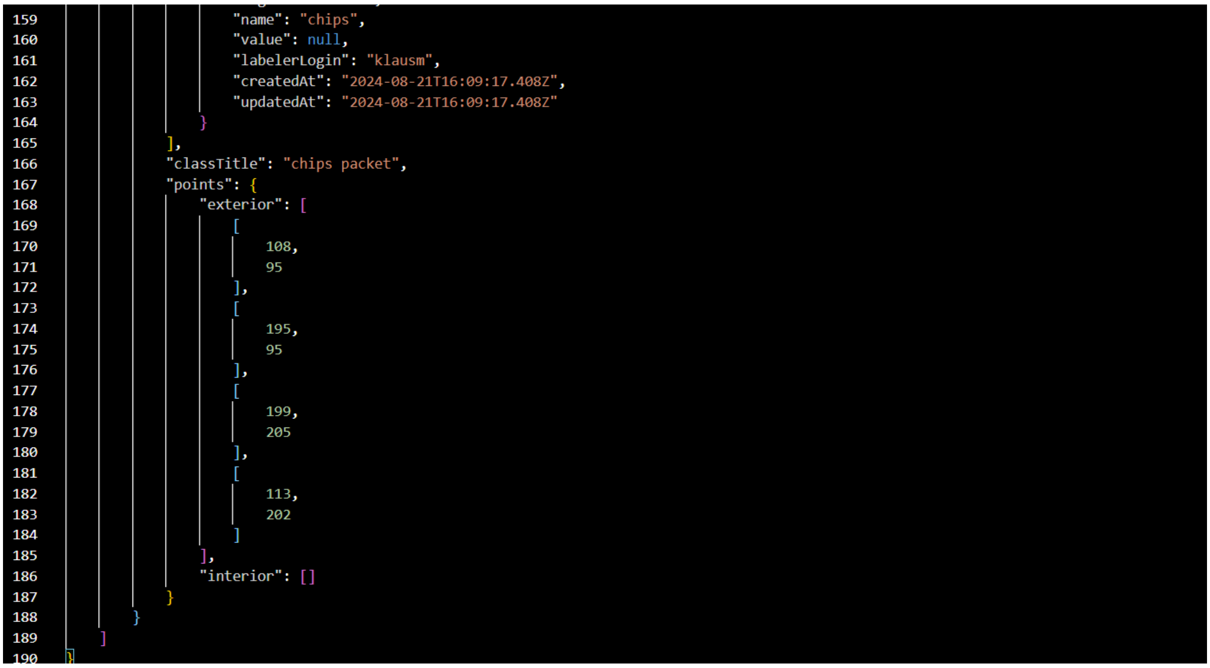
```
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T13:12:57.276Z",
            "updatedAt": "2024-08-21T13:13:00.516Z",
            "tags": [
                {
                    "id": 228584018,
                    "tagId": 32752199,
                    "name": "chips",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:09:23.694Z",
                    "updatedAt": "2024-08-21T16:09:23.694Z"
                }
            ],
            "classTitle": "chips packet",
            "points": {
                "exterior": [
                    [
                        118,
                        9
                    ],
                    [
                        211,
                        23
                    ],
                    [
                        192,
                        91
                    ],
                    [
                        113,
```



```
                        92
                    ]
                ],
                "interior": []
            }
        },
        {
            "id": 1586577796,
            "classId": 13296645,
            "objectId": null,
            "description": "",
            "geometryType": "polygon",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T13:13:14.371Z",
            "updatedAt": "2024-08-21T13:13:14.371Z",
            "tags": [
                {
                    "id": 228584017,
                    "tagId": 32752199,
                    "name": "chips",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:09:20.639Z",
                    "updatedAt": "2024-08-21T16:09:20.639Z"
                }
            ],
            "classTitle": "chips packet",
            "points": {
                "exterior": [
                    [
                        26,
```



```
                        95
                    ],
                    [
                        103,
                        95
                    ],
                    [
                        110,
                        204
                    ],
                    [
                        25,
                        203
                    ]
                ],
                "interior": []
            }
        },
        {
            "id": 1586577801,
            "classId": 13296645,
            "objectId": null,
            "description": "",
            "geometryType": "polygon",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T13:13:29.765Z",
            "updatedAt": "2024-08-21T13:13:29.765Z",
            "tags": [
                {
                    "id": 228584016,
                    "tagId": 32752199,
```



```
                    "name": "chips",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:09:17.408Z",
                    "updatedAt": "2024-08-21T16:09:17.408Z"
                }
            ],
            "classTitle": "chips packet",
            "points": {
                "exterior": [
                    [
                        108,
                        95
                    ],
                    [
                        195,
                        95
                    ],
                    [
                        199,
                        205
                    ],
                    [
                        113,
                        202
                    ]
                ],
                "interior": []
            }
        }
    ]
}
```

IMAGE -2

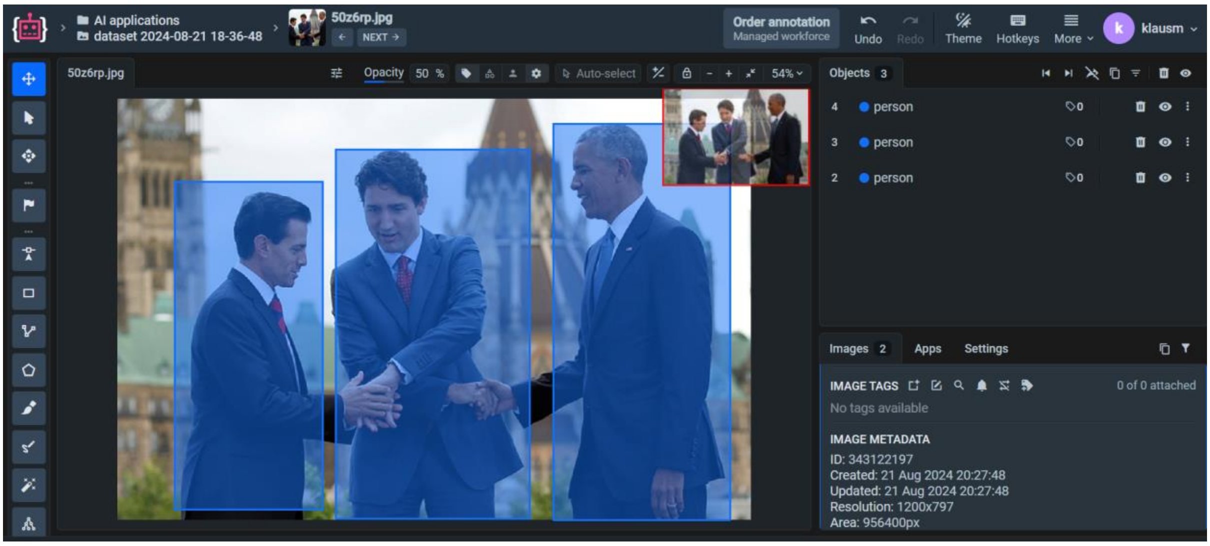
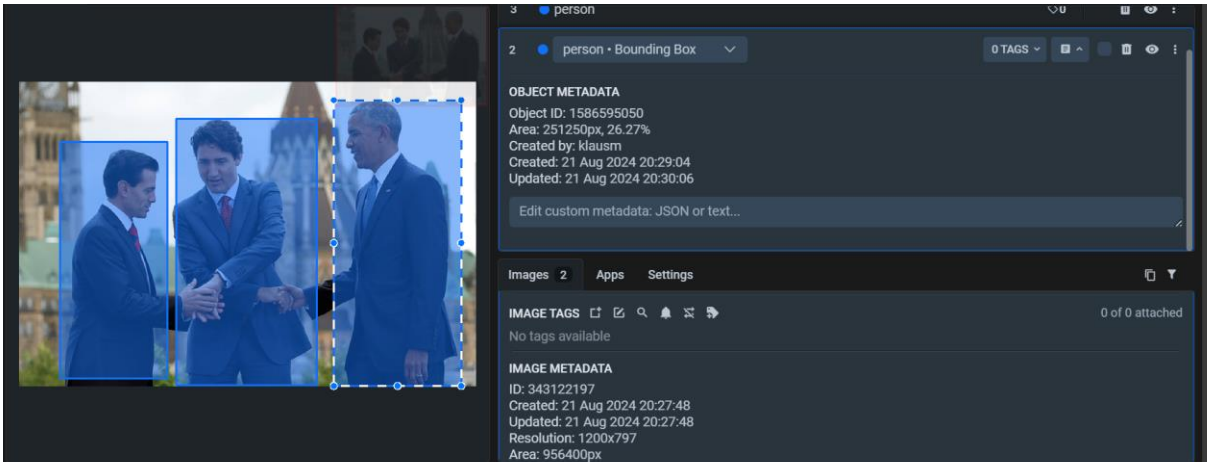


IMAGE TAGS 0 of 0 attached

No tags available

IMAGE METADATA
ID: 343122197
Created: 21 Aug 2024 20:27:48
Updated: 21 Aug 2024 20:27:48
Resolution: 1200x797
Area: 956400px



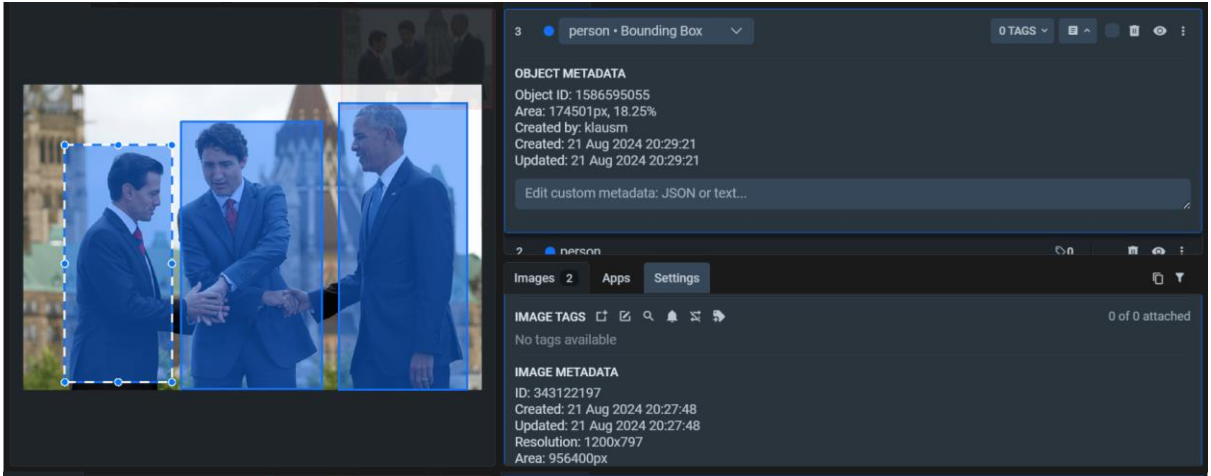
OBJECT METADATA
Object ID: 1586595050
Area: 251250px, 26.27%
Created by: klausm
Created: 21 Aug 2024 20:29:04
Updated: 21 Aug 2024 20:30:06

Edit custom metadata: JSON or text...

IMAGE TAGS 0 of 0 attached

No tags available

IMAGE METADATA
ID: 343122197
Created: 21 Aug 2024 20:27:48
Updated: 21 Aug 2024 20:27:48
Resolution: 1200x797
Area: 956400px



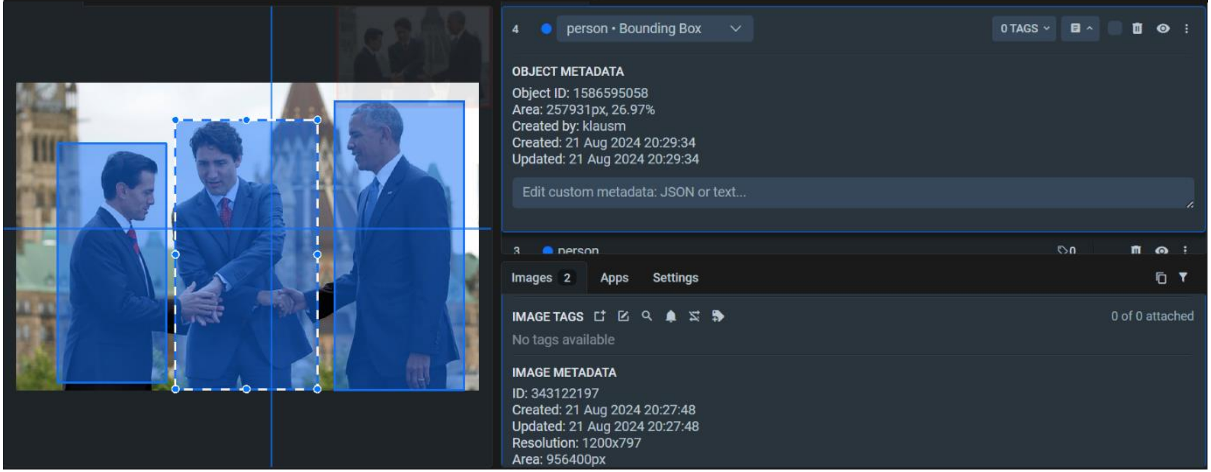
OBJECT METADATA
Object ID: 1586595055
Area: 174501px, 18.25%
Created by: klausm
Created: 21 Aug 2024 20:29:21
Updated: 21 Aug 2024 20:29:21

Edit custom metadata: JSON or text...

IMAGE TAGS 0 of 0 attached

No tags available

IMAGE METADATA
ID: 343122197
Created: 21 Aug 2024 20:27:48
Updated: 21 Aug 2024 20:27:48
Resolution: 1200x797
Area: 956400px



OBJECT METADATA
Object ID: 1586595058
Area: 257931px, 26.97%
Created by: klausm
Created: 21 Aug 2024 20:29:34
Updated: 21 Aug 2024 20:29:34

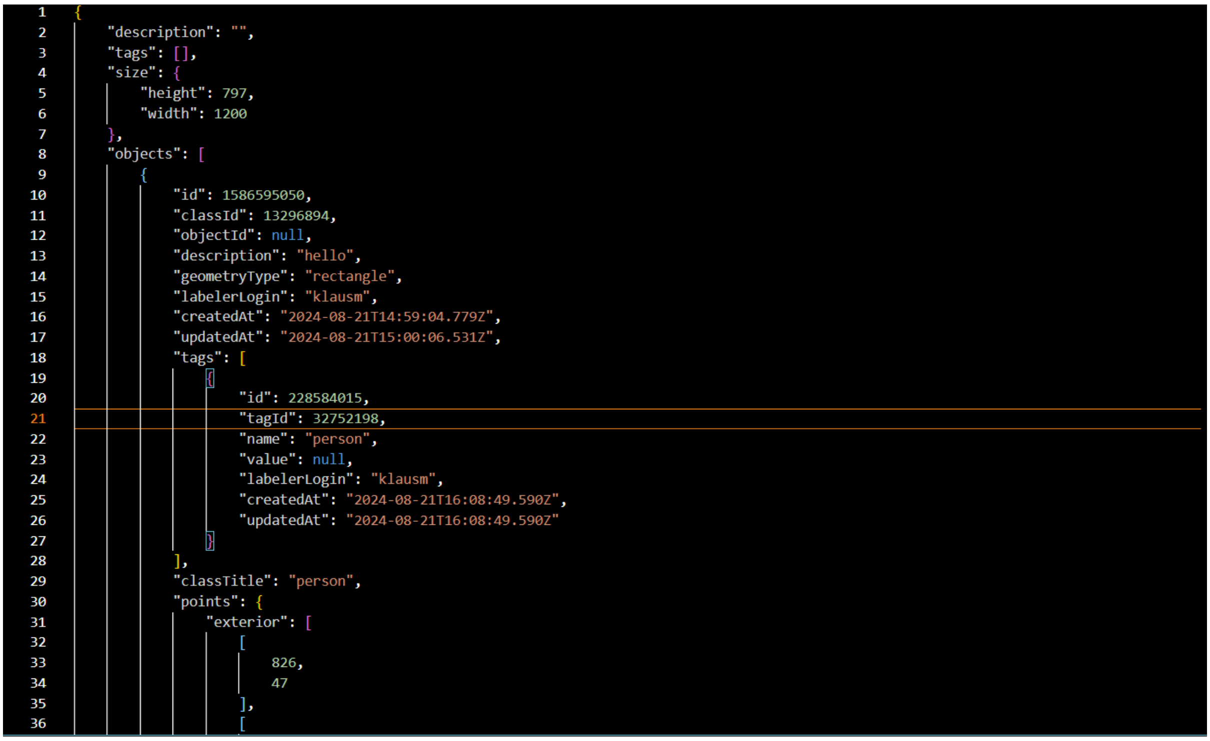
Edit custom metadata: JSON or text...

IMAGE TAGS 0 of 0 attached

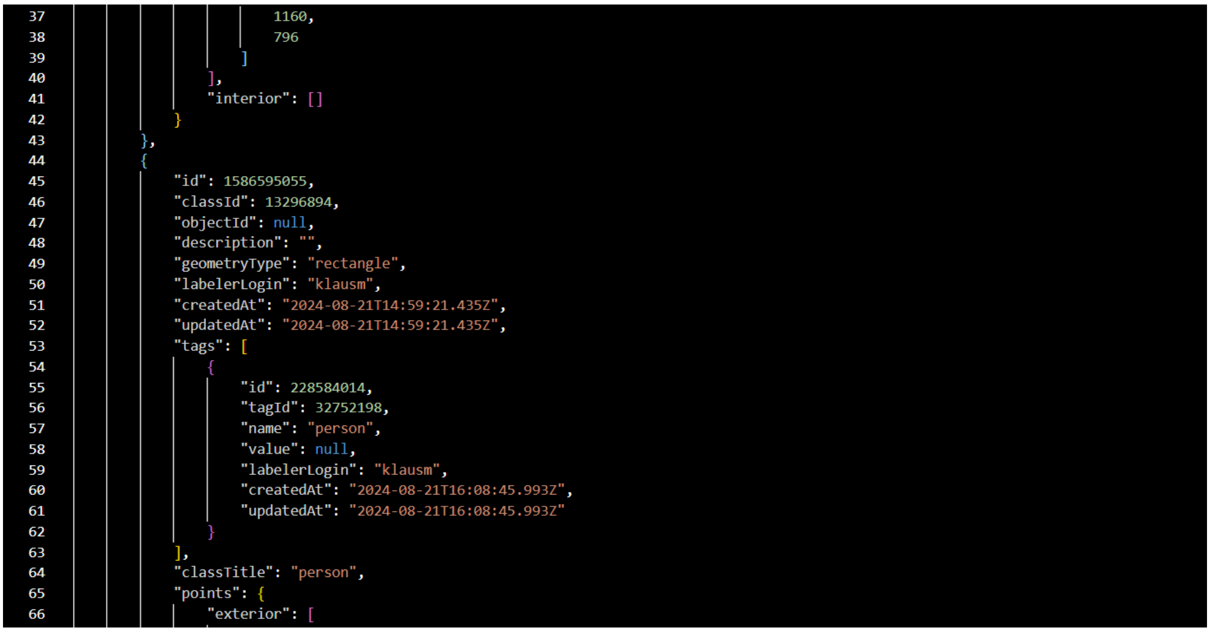
No tags available

IMAGE METADATA
ID: 343122197
Created: 21 Aug 2024 20:27:48
Updated: 21 Aug 2024 20:27:48
Resolution: 1200x797
Area: 956400px

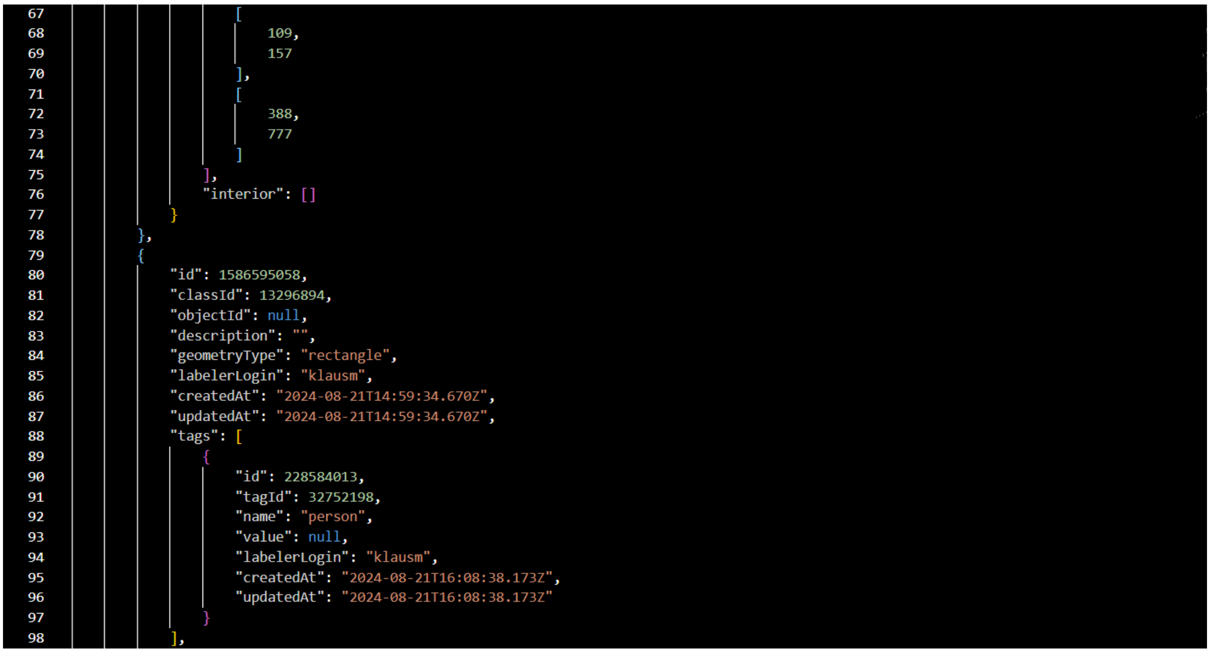
IMAGE 2 JSON:



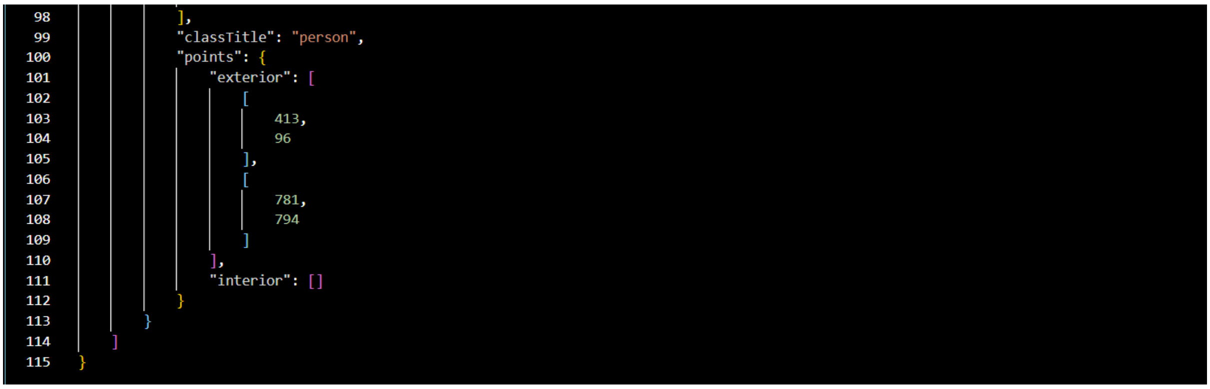
```
{
    "description": "",
    "tags": [],
    "size": {
        "height": 797,
        "width": 1200
    },
    "objects": [
        {
            "id": 1586595050,
            "classId": 13296894,
            "objectId": null,
            "description": "hello",
            "geometryType": "rectangle",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T14:59:04.779Z",
            "updatedAt": "2024-08-21T15:00:06.531Z",
            "tags": [
                {
                    "id": 228584015,
                    "tagId": 32752198,
                    "name": "person",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:08:49.590Z",
                    "updatedAt": "2024-08-21T16:08:49.590Z"
                }
            ],
            "classTitle": "person",
            "points": {
                "exterior": [
                    [
                        826,
                        47
                    ],
                    [
```



```
                        1160,
                        796
                    ]
                ],
                "interior": []
            }
        },
        {
            "id": 1586595055,
            "classId": 13296894,
            "objectId": null,
            "description": "",
            "geometryType": "rectangle",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T14:59:21.435Z",
            "updatedAt": "2024-08-21T14:59:21.435Z",
            "tags": [
                {
                    "id": 228584014,
                    "tagId": 32752198,
                    "name": "person",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:08:45.993Z",
                    "updatedAt": "2024-08-21T16:08:45.993Z"
                }
            ],
            "classTitle": "person",
            "points": {
                "exterior": [
```

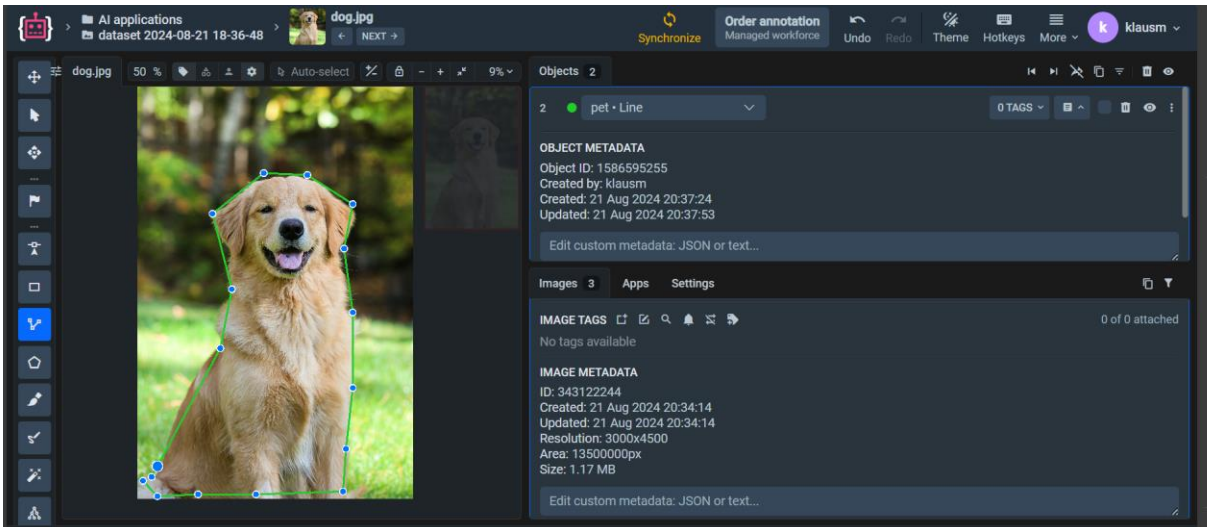


```
                    [
                        109,
                        157
                    ],
                    [
                        388,
                        777
                    ]
                ],
                "interior": []
            }
        },
        {
            "id": 1586595058,
            "classId": 13296894,
            "objectId": null,
            "description": "",
            "geometryType": "rectangle",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T14:59:34.670Z",
            "updatedAt": "2024-08-21T14:59:34.670Z",
            "tags": [
                {
                    "id": 228584013,
                    "tagId": 32752198,
                    "name": "person",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:08:38.173Z",
                    "updatedAt": "2024-08-21T16:08:38.173Z"
                }
            ],
```



```
            ],
            "classTitle": "person",
            "points": {
                "exterior": [
                    [
                        413,
                        96
                    ],
                    [
                        781,
                        794
                    ]
                ],
                "interior": []
            }
        }
    ]
}
```

IMAGE-3



OBJECT METADATA
Object ID: 1586595255
Created by: klausm
Created: 21 Aug 2024 20:37:24
Updated: 21 Aug 2024 20:37:53

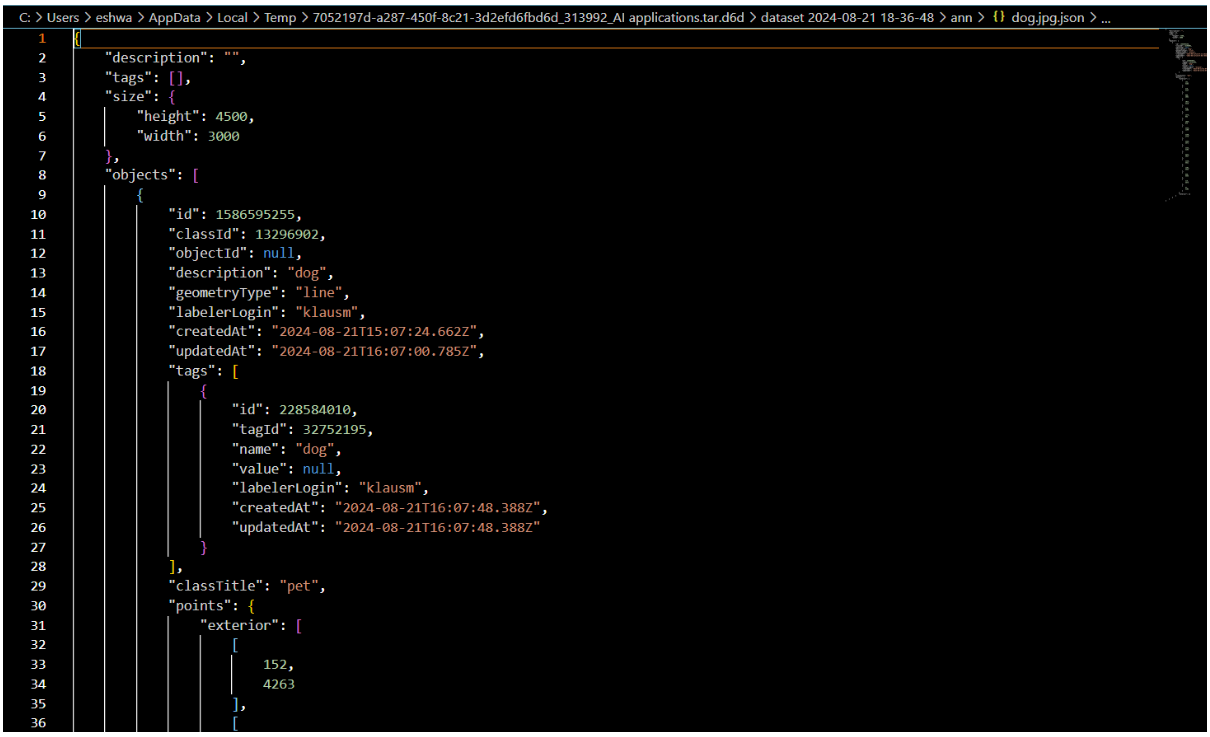
Edit custom metadata: JSON or text...

IMAGE TAGS
No tags available

IMAGE METADATA
ID: 343122244
Created: 21 Aug 2024 20:34:14
Updated: 21 Aug 2024 20:34:14
Resolution: 3000x4500
Area: 13500000px
Size: 1.17 MB

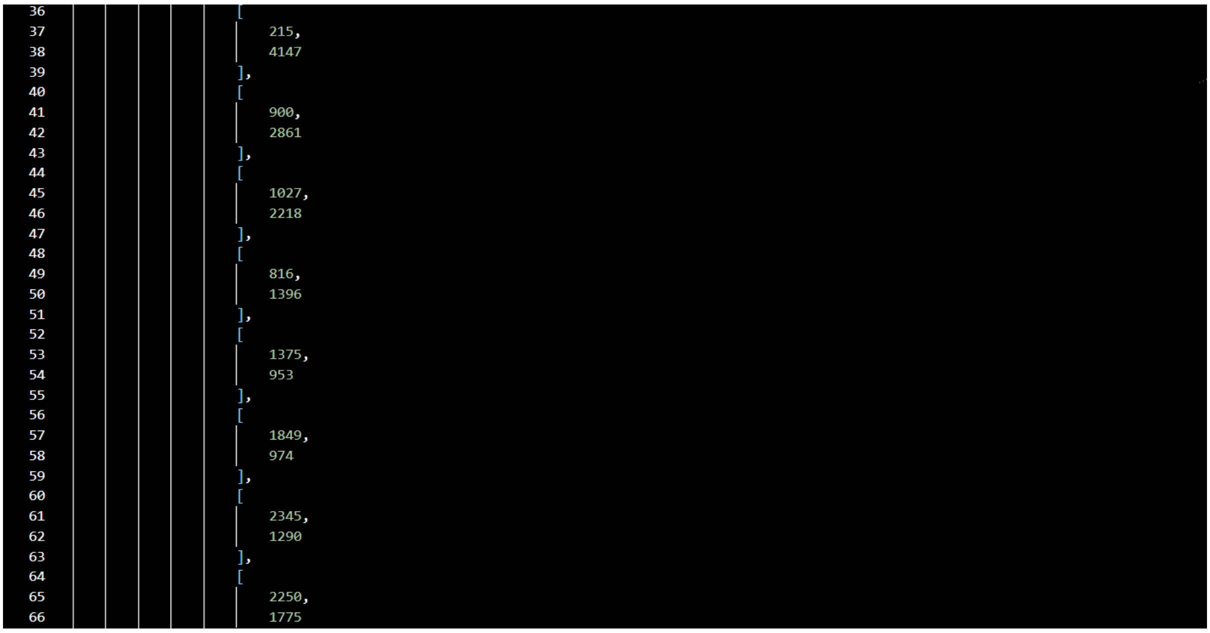
Edit custom metadata: JSON or text...

IMAGE 3 JSON:

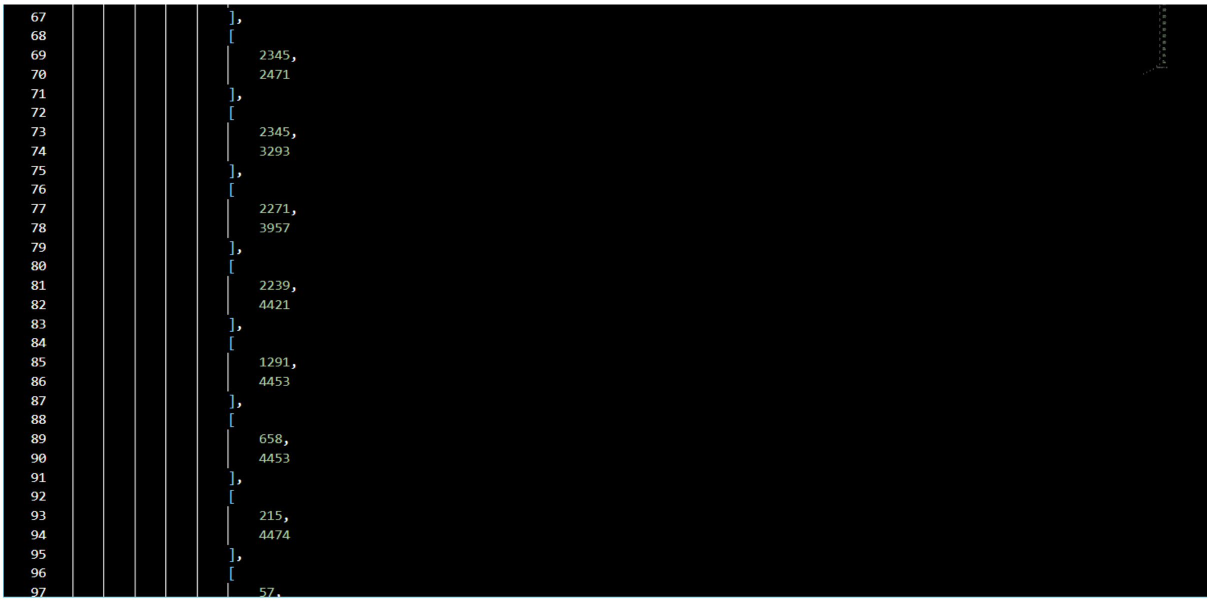


C: > Users > eshwa > AppData > Local > Temp > 7052197d-a287-450f-8c21-3d2efd6fbd6d_313992_AI applications.tar.d6d > dataset 2024-08-21 18-36-48 > ann > {} dog.jpg.json > ...

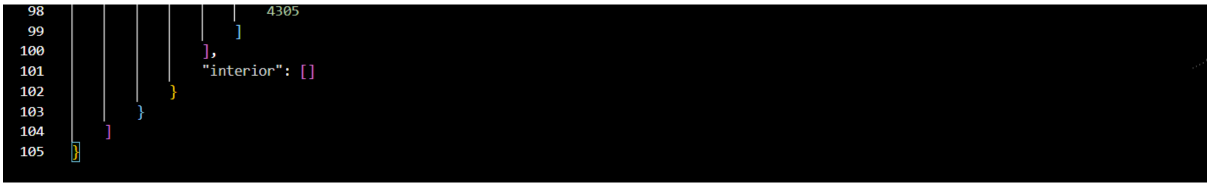
```
{
    "description": "",
    "tags": [],
    "size": {
        "height": 4500,
        "width": 3000
    },
    "objects": [
        {
            "id": 1586595255,
            "classId": 13296902,
            "objectId": null,
            "description": "dog",
            "geometryType": "line",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T15:07:24.662Z",
            "updatedAt": "2024-08-21T16:07:00.785Z",
            "tags": [
                {
                    "id": 228584010,
                    "tagId": 32752195,
                    "name": "dog",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:07:48.388Z",
                    "updatedAt": "2024-08-21T16:07:48.388Z"
                }
            ],
            "classTitle": "pet",
            "points": {
                "exterior": [
                    [
                        152,
                        4263
                    ],
                    [
```



```
                    [
                        215,
                        4147
                    ],
                    [
                        900,
                        2861
                    ],
                    [
                        1027,
                        2218
                    ],
                    [
                        816,
                        1396
                    ],
                    [
                        1375,
                        953
                    ],
                    [
                        1849,
                        974
                    ],
                    [
                        2345,
                        1290
                    ],
                    [
                        2250,
                        1775
```

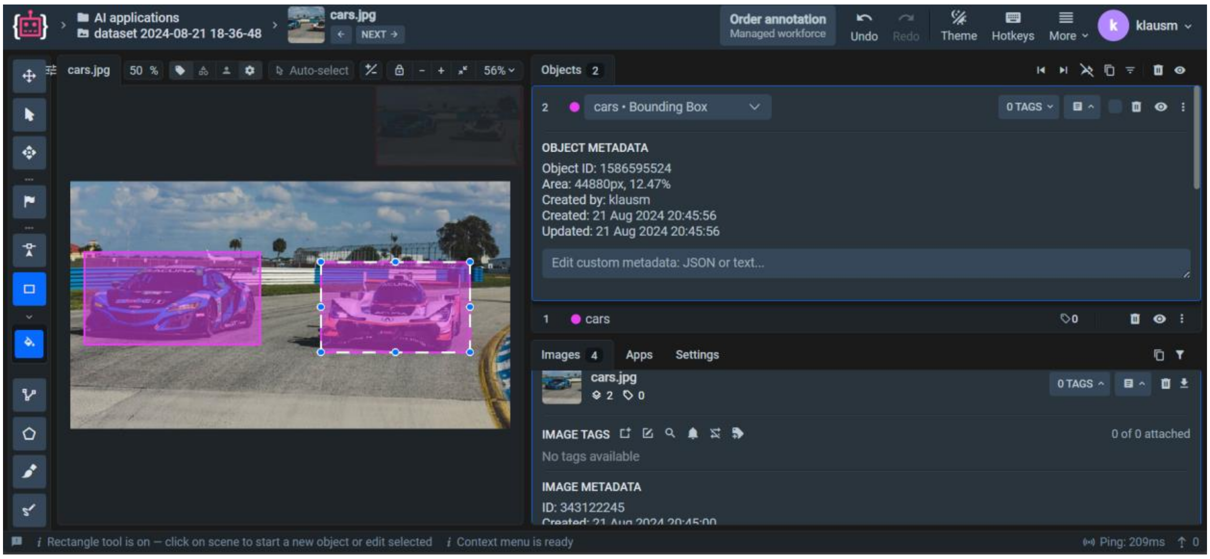


```
                        ],
                        [
                            2345,
                            2471
                        ],
                        [
                            2345,
                            3293
                        ],
                        [
                            2271,
                            3957
                        ],
                        [
                            2239,
                            4421
                        ],
                        [
                            1291,
                            4453
                        ],
                        [
                            658,
                            4453
                        ],
                        [
                            215,
                            4474
                        ],
                        [
                            57,
```



```
                            4305
                        ]
                    ],
                    "interior": []
                }
            }
        ]
}
```

IMAGE-4



Objects 2

2 cars • Bounding Box 0 TAGS

OBJECT METADATA
Object ID: 1586595524
Area: 44880px, 12.47%
Created by: klausm
Created: 21 Aug 2024 20:45:56
Updated: 21 Aug 2024 20:45:56

Edit custom metadata: JSON or text...

1 cars 0

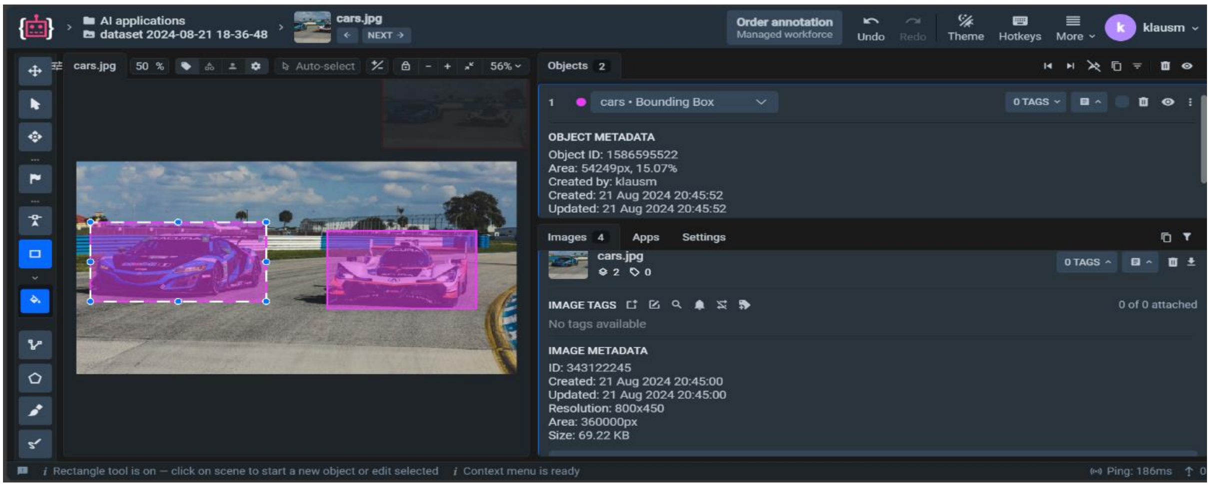
Images 4 Apps Settings

cars.jpg 2 0 0 TAGS

IMAGE TAGS 0 of 0 attached
No tags available

IMAGE METADATA
ID: 343122245
Created: 21 Aug 2024 20:45:00

Rectangle tool is on — click on scene to start a new object or edit selected   Context menu is ready   Ping: 209ms 0



Objects 2

1 cars • Bounding Box 0 TAGS

OBJECT METADATA
Object ID: 1586595522
Area: 54249px, 15.07%
Created by: klausm
Created: 21 Aug 2024 20:45:52
Updated: 21 Aug 2024 20:45:52

Images 4 Apps Settings

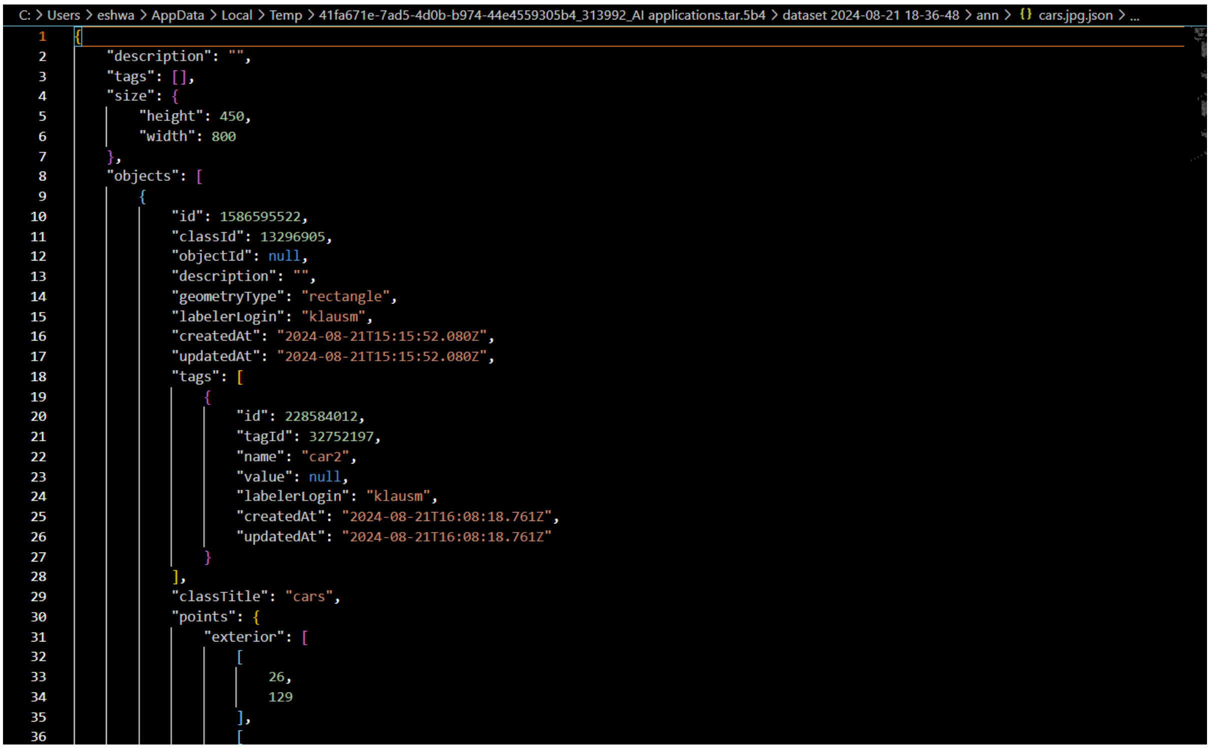
cars.jpg 2 0 0 TAGS

IMAGE TAGS 0 of 0 attached
No tags available

IMAGE METADATA
ID: 343122245
Created: 21 Aug 2024 20:45:00
Updated: 21 Aug 2024 20:45:00
Resolution: 800x450
Area: 360000px
Size: 69.22 KB

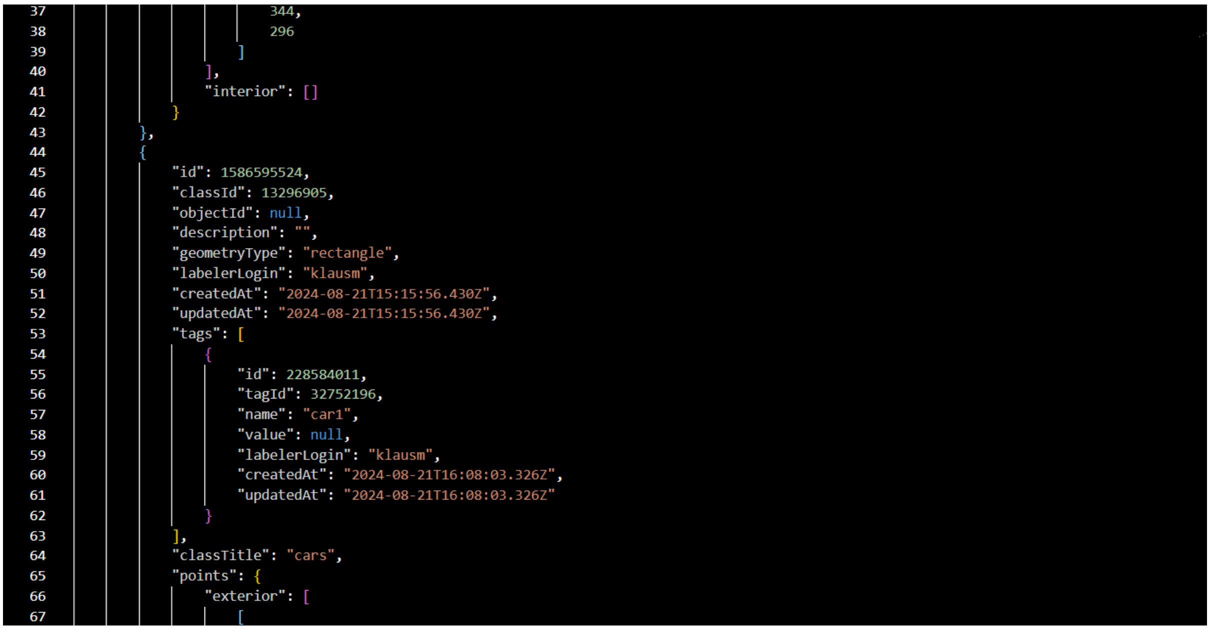
Rectangle tool is on — click on scene to start a new object or edit selected   Context menu is ready   Ping: 186ms 0

IMAGE 4 JSON:

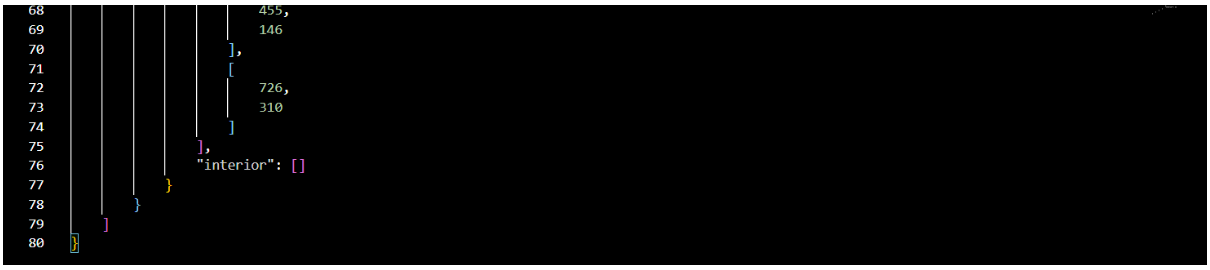


C: > Users > eshwa > AppData > Local > Temp > 41fa671e-7ad5-4d0b-b974-44e4559305b4_313992_AI applications.tar.5b4 > dataset 2024-08-21 18-36-48 > ann > {} cars.jpg.json > ...

```
{
    "description": "",
    "tags": [],
    "size": {
        "height": 450,
        "width": 800
    },
    "objects": [
        {
            "id": 1586595522,
            "classId": 13296905,
            "objectId": null,
            "description": "",
            "geometryType": "rectangle",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T15:15:52.080Z",
            "updatedAt": "2024-08-21T15:15:52.080Z",
            "tags": [
                {
                    "id": 228584012,
                    "tagId": 32752197,
                    "name": "car2",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:08:18.761Z",
                    "updatedAt": "2024-08-21T16:08:18.761Z"
                }
            ],
            "classTitle": "cars",
            "points": {
                "exterior": [
                    [
                        26,
                        129
                    ],
                    [
```



```
                        344,
                        296
                    ]
                ],
                "interior": []
            }
        },
        {
            "id": 1586595524,
            "classId": 13296905,
            "objectId": null,
            "description": "",
            "geometryType": "rectangle",
            "labelerLogin": "klausm",
            "createdAt": "2024-08-21T15:15:56.430Z",
            "updatedAt": "2024-08-21T15:15:56.430Z",
            "tags": [
                {
                    "id": 228584011,
                    "tagId": 32752196,
                    "name": "car1",
                    "value": null,
                    "labelerLogin": "klausm",
                    "createdAt": "2024-08-21T16:08:03.326Z",
                    "updatedAt": "2024-08-21T16:08:03.326Z"
                }
            ],
            "classTitle": "cars",
            "points": {
                "exterior": [
                    [
```



```
                        455,
                        146
                    ],
                    [
                        726,
                        310
                    ]
                ],
                "interior": []
            }
        }
    ]
}
```