Assessment Task: Vision Record Generation

Objective:
Your task is to create a Python function that takes an image as input and outputs a vision record. This record should include details about objects detected in the image, their activities, colors, bounding boxes, frame size, and a summary of the frame.

Important Note: Please ensure that your solution is based on open-source models.
You are not allowed to use GPT Vision for this task.

Function Signature:

```
def generate_vision_record(image: np.array) -> dict:
    """
    Generates a vision record from the input image.

    Parameters:
        image (np.array): The input image as a NumPy array.

    Returns:
        dict: A dictionary containing the vision record.
    """
```

Expected Output:

The function should output a dictionary with the following structure:

```
{
    "Time": "YYYY-MM-DD HH:MM:SS.ssssss",  # Timestamp of when the image is processed
    "Objects": ["vehicle", "vehicle", "person"],  # List of detected objects
    "Objects Activities": [None, None, "Standing"],  # Activities of the detected objects
    "Object Colors": ["black", "red", "red"],  # Colors of the detected objects
```

```
    "Object Bounding Boxes": [[390, 105, 472, 148], [305, 95, 400, 100], [315, 105, 410, 110]],  # Bounding boxes of the detected objects (x, y, w, h)
    "Frame Size": (480, 640),  # Size of the frame in pixels (height, width)
    "Frame Summary": "A man wearing a red shirt leaving the apartment with an object which is a jerrican"  # Text summary of the frame
}
```

Requirements:

1. **Object Detection**: Implement a method to detect objects within the image and classify them into categories like "vehicle", "person", etc.

2. **Activity Recognition**: For each detected object, identify if there's any associated activity (e.g., standing, walking, sitting). If no activity is detected, return `None`.

3. **Color Detection**: Identify the primary color of each detected object.

4. **Bounding Boxes**: Return the bounding box coordinates for each detected object in the format `[x, y, w, h]` where `(x, y)` represents the top-left corner of the box, and `(w, h)` represents the width and height.

5. **Frame Size**: Extract and return the size of the image frame in pixels `(height, width)`.

6. **Frame Summary**: Generate a textual summary of the frame, describing the main event or action taking place in the image.

7. **Timestamp**: The time should be captured at the moment the image is processed.

Evaluation Criteria:

- **Accuracy**: The accuracy of object detection, activity recognition, and color identification.
- **Efficiency**: The function should be optimized for performance.

- **Code Quality**: Clean, well-documented, and maintainable code.
- **Creativity**: Innovative approaches to solving the problem.

Tools and Libraries:

You may use any Python libraries such as OpenCV, TensorFlow, PyTorch, etc., to implement this function.

Submission:

Submit your code in a `.py` file or Jupyter notebook. Make sure to include instructions for running the code and any necessary dependencies. The due date for this assessment is 12th August 2024.