



[Click Here to Subscribe to my YouTube Channel >>](#)

100% Automated Youtube Shorts and Instagram Reels (AI + Make) - Prompts & Links

(Follow the instructions on this video: <https://youtu.be/iBAzffEMYwc>)

Links:

Create a free Make account:

<https://www.make.com/>

Set up a free Google Account here (if you don't have one yet):

<https://accounts.google.com/>

Create OpenAI account to get API:

<https://auth.openai.com/authorize>

Create Creatomate account to get API:

<https://creatomate.com>

Create Pexels account to get API:

<https://www.pexels.com/de-de/api/key/>

<https://api.pexels.com/videos/search>

<https://www.pexels.com/download/video/{{97.`Random Video`.id}}/>

Create Elevenlabs account to get API:

<https://elevenlabs.io/>

Create Twitter/X developer account to get API/Client ID:

<https://developer.twitter.com/>

Google Sheet Template:

<https://docs.google.com/spreadsheets/d/1ULsr3VRg15YefrcbNU7S3aXvmKjoGRCDs-ay1eYH9o0/copy>

Prompts:

Topic Generation Prompt:

Come up with ONE specific topic that this audience and/or niche is likely very interested in. It should be a subtopic of the overarching niche or audience, or processes broken down that are required to get a certain result or outcome or solve a certain problem. The best topics are usually related to a specific problem the audience has, a specific result the audience wants to achieve within the subtopic.

This is the niche/audience: [variable niche/topic]

Your topic must be exactly based upon the the niche/audience mentioned but you are not allowed to use any of these subtopics which have already been covered:

[variable from text aggregator]

#Rule: You must keep our response below 300 characters.

Transcript Generator Prompt:

Behave like a professional Instagram reel scripts copywriter that specializes in writing viral Instagram reel scripts in English that get lots of followers, likes and shares. I need a highly viral and engaging Instagram caption that captivates my audience.

The topic of the reel is:

[subtopic variable]

The reel script must be EXACTLY based on the structure this script that has worked well:

[transcript template variable]

Follow the structure and angle/strategy that makes it so captivating of the example post as EXACT as possible and only adjust the transcript example to the topic and it must have the same number of sentences as the example transcript, not more and not less.

No emojis. No hashtags. One line break between each sentence. No formatting such as bolding of the text of the script. Before every new sentence and every new line you MUST type "/".

Only give the raw script text, do never say things like "Here's the reel script."

Caption Generator Prompt:

Behave like a professional Instagram captions copywriter that specialises in writing viral Instagram captions in English that get lots of followers, likes and shares. I need a highly viral and engaging Instagram caption that captivates my audience.

The topic of the reel is:

[subtopic variable]

The reel script must be EXACTLY based on the structure these captions that have worked well:

[captions template variable]

Follow the structure and angle/strategy that makes it so captivating of the example post as EXACT as possible and only adjust the caption example to the topic and it must have the same number of sentences. One line break between each sentence. No formatting such as bolding of the text of the script.

Only give the raw captions text, do never say things like "Here's the reel script."

Google App Script:

Make sure to follow the exact instructions in the video tutorial so you know how to properly set up this app script.

```
function rotateAndSendData() {
  // Open the spreadsheet and get the 'Template' and 'Topics' sheets
  var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('Template');
  var topicsSheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName('Input');
  var lastRow = sheet.getLastRow();
  var lastTopicRow = topicsSheet.getLastRow(); // Get the last row in the
Topics sheet

  // Get the current row number for templates (stored in script properties)
  var rowToSend =
PropertiesService.getScriptProperties().getProperty('currentRow') || 2; //
Start with row 2

  // Convert the row number to an integer
  rowToSend = parseInt(rowToSend, 10);

  // Get the current topic index (starting from row 2 in the 'Topics'
sheet)
  var topicIndex =
PropertiesService.getScriptProperties().getProperty('currentTopicIndex') ||
2; // Start at row 2 for topics

  // Convert topic index to an integer
  topicIndex = parseInt(topicIndex, 10);

  // Get the current topic
  var topic = topicsSheet.getRange(topicIndex, 1).getValue(); // Column A
contains the topics

  // Get the data from the current row in the 'Template' sheet (e.g.,
transcript and captions)
  var transcript = sheet.getRange(rowToSend, 2).getValue();
  var captions = sheet.getRange(rowToSend, 3).getValue();

  // Send the data to a webhook URL, including the current topic
  var url = 'INSERT-YOUR-MAKE-HOOK'; // Replace with your actual webhook
URL
  var payload = {
    "transcript": transcript,
```

```

    "captions": captions,
    "topic": topic // Send the current topic with the post
  };

  var options = {
    'method': 'post',
    'contentType': 'application/json',
    'payload': JSON.stringify(payload)
  };

  UrlFetchApp.fetch(url, options);

  // Update the current row number for templates, loop back to row 2 if we
  hit the last row
  if (rowToSend < lastRow) {
    PropertiesService.getScriptProperties().setProperty('currentRow',
(rowToSend + 1).toString());
  } else {
    PropertiesService.getScriptProperties().setProperty('currentRow', '2');
  // Loop back to row 2
  }

  // Now check if we've rotated through 4 posts (1 cycle of 4 templates)
  var postsCount =
PropertiesService.getScriptProperties().getProperty('postsCount') || 0; //
Track number of posts
  postsCount = parseInt(postsCount, 10) + 1; // Increment post count

  if (postsCount >= 4) {
    // Reset post count to 0 after every 4 posts
    PropertiesService.getScriptProperties().setProperty('postsCount', 0);

    // Move to the next topic, loop back to the first topic if we've
reached the end
    if (topicIndex < lastTopicRow) {

PropertiesService.getScriptProperties().setProperty('currentTopicIndex',
(topicIndex + 1).toString());
    } else {

PropertiesService.getScriptProperties().setProperty('currentTopicIndex',
'2'); // Loop back to the first topic
    }
  } else {
    // If we haven't reached 4 posts yet, just update the post count
    PropertiesService.getScriptProperties().setProperty('postsCount',
postsCount.toString());
  }

```

```
}  
}
```

```
function processImagesInAllSheets() {  
    var sheetNames = ["Video"];  
    var spreadsheet = SpreadsheetApp.getActiveSpreadsheet();  
  
    sheetNames.forEach(function(sheetName) {  
        var sheet = spreadsheet.getSheetByName(sheetName);  
        if (sheet) {  
            Logger.log('Processing sheet: ' + sheetName);  
            var lastRow = sheet.getLastRow();  
  
            for (var row = 2; row <= lastRow; row++) { // Start from row 2 to  
skip headers  
                var urlCell = sheet.getRange(row, 3); // Column B  
                var imageCell = sheet.getRange(row, 4); // Column C  
  
                var url = urlCell.getValue();  
                Logger.log('Row ' + row + ': URL = ' + url);  
  
                if (url && imageCell.getFormula() === "" && imageCell.getValue()  
=== "") {  
                    Logger.log('Inserting image at row ' + row);  
                    imageCell.setFormula('=IMAGE("' + url + '", 1)');  
                    sheet.setRowHeight(row, 200);  
                }  
            }  
  
            // Set the column width for images  
            sheet.setColumnWidth(4, 200); // Column C is the 3rd column  
        } else {  
            Logger.log('Sheet ' + sheetName + ' not found!');  
        }  
    });  
}
```

```
function onChange(e) {  
    // First, run the email sending function exactly as it is  
    sendEmail();  
}
```

```

    // Then, run the image insertion function exactly as it is
    insertImage();
}

// The original email function without any changes
function sendEmail() {
    var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName("Video");

    // Get the entire range of column B
    var dataRange = sheet.getRange(2, 2, sheet.getLastRow() - 1, 1); // B2 to
last row
    var currentData = dataRange.getValues(); // Get current values of column
B

    // Get the stored previous values from the properties service
    var scriptProperties = PropertiesService.getScriptProperties();
    var previousData =
JSON.parse(scriptProperties.getProperty("previousData") || "[]");

    // Loop through the rows and check for new entries
    for (var i = 0; i < currentData.length; i++) {
        var newValue = currentData[i][0];
        var oldValue = previousData[i] ? previousData[i][0] : "";

        // Check if there is a new value in column B that starts with 'http'
        if (newValue && newValue.startsWith("http") && newValue !== oldValue) {
            var email = "INSERT-YOUR-EMAIL"; // Email address of the recipient
            var subject = "TikTok Video ready for upload";
            var message = "In the attachment, you'll find the video ready for
TikTok upload.";

            try {
                // Download the video from the provided link
                var response = UrlFetchApp.fetch(newValue);
                var blob = response.getBlob();

                // Send the email with the video attachment
                MailApp.sendEmail({
                    to: email,
                    subject: subject,
                    body: message,
                    attachments: [blob]
                });

                Logger.log("Email sent successfully for video link: " + newValue);
            }
        }
    }
}

```

```

    } catch (error) {
        Logger.log("Failed to send email for video link: " + newValue + ".
Error: " + error.message);
    }
}
}

// Store the current data for future comparison
scriptProperties.setProperty("previousData",
JSON.stringify(currentData));
}

// The original image insertion function without any changes
function insertImage() {
    var sheet =
SpreadsheetApp.getActiveSpreadsheet().getSheetByName("Video");

    // Get the entire range of column C
    var dataRange = sheet.getRange(2, 3, sheet.getLastRow() - 1, 1); // C2 to
last row
    var currentData = dataRange.getValues(); // Get current values of column
C

    // Get the stored previous values from the properties service for column
C
    var scriptProperties = PropertiesService.getScriptProperties();
    var previousData =
JSON.parse(scriptProperties.getProperty("previousImageURLs") || "[]");

    // Loop through the rows and check for new entries in column C (image
URLs)
    for (var i = 0; i < currentData.length; i++) {
        var newValue = currentData[i][0];
        var oldValue = previousData[i] ? previousData[i][0] : "";

        // Check if there is a new value in column C that starts with 'http'
        if (newValue && newValue.startsWith("http") && newValue !== oldValue) {
            var imageCell = sheet.getRange(i + 2, 4); // Get the corresponding
cell in column D

            // Insert the image in column D
            if (imageCell.getFormula() === "" && imageCell.getValue() === "") {
                try {
                    Logger.log("Inserting image for URL: " + newValue + " at row " +
(i + 2));
                    imageCell.setFormula('=IMAGE("'" + newValue + "', 1)');

```



```
        sheet.setRowHeight(i + 2, 200); // Adjust row height for the
image
        sheet.setColumnWidth(4, 200); // Adjust the width of column D to
fit the image
    } catch (error) {
        Logger.log("Error inserting image: " + error.message);
    }
}
}
}

// Store the current data of column C for future comparison
scriptProperties.setProperty("previousImageURLs",
JSON.stringify(currentData));
}
```