Date 6/9/2024

Project Summary: Auto-Like Tweets Chrome Extension for X.com (Twitter)

Client Requirement:

Here are the client's requirements for the auto-liker in their own words:

1. Auto Liker Chrome Extension:
    a. "Can you make me that chrome extension to like the tweets?"
2. Liking Feature:
    b. "With the liking feature how fast will it be liking tweets, will I be able to set a maximum liking amount for every 1hr for example?"
    c. "Also will it just start liking tweets from the timeline which is currently presented, how does this work? For example, if I'm in the likes tab section of an account and I choose 50 likes, will it just like those tweets in the likes tab section until it has liked 50 tweets?"
    d. "Also if it has already liked the tweet, will it be able to identify this and not click the like button if the tweet is already liked?"

Extension Features:

1. X.com Verification:
    a. The extension first checks if X.com is open in the current browser tab.
    b. If X.com is not open, a message is displayed to the user, asking them to open X.com for the extension to function properly.
2. Tweet Filtering:
    a. The extension scans the page for tweets, excluding tweets that are already liked and reply posts.
    b. This ensures that the extension only targets new, unliked tweets.

Functionalities:

Auto-Liking:

1. The extension automatically likes the filtered tweets.
2. After liking a tweet, it waits for 2 seconds before changing the tweet's background to green and hiding it from the page.
3. This visual feedback helps the user understand which tweets have been processed.

Real-Time Reporting:

1. The extension logs the number of tweets skipped (already liked) and the number of tweets successfully liked.

2. A report log is shown to the user in the extension's interface, keeping them informed about the extension's activity.

Error Handling:

1. The extension handles various errors, including network problems and issues loading new posts.
2. Any errors encountered during execution are logged to the console.
3. The extension ensures that it continues to function normally despite these errors, providing a seamless user experience.

Development Approach:

- Utilized JavaScript and Chrome Extension APIs to create the extension.
- Implemented DOM manipulation techniques to interact with and modify tweet elements on the page.
- Added robust error handling mechanisms to manage network issues and execution errors.
- Created an intuitive user interface for real-time reporting and user feedback.

Challenges and Solutions:

- Network Issues: Implemented retry logic and error handling to manage scenarios where tweets fail to load due to network problems.
- DOM Manipulation: Ensured accurate selection and modification of tweet elements, avoiding interference with the normal user experience on X.com.
- User Feedback: Designed a clear and informative reporting system within the extension to keep the user informed about the extension's actions and status.

Conclusion:

The auto-like tweets Chrome extension for X.com successfully meets the client's requirements. It provides a reliable and user-friendly solution for automatically liking tweets while handling various errors gracefully and keeping the user informed through real-time reporting. This extension enhances the user's experience on X.com by automating the liking process efficiently and effectively.

Flow of the Tweet Auto Liker Chrome Extension

Introduction

The Tweet Auto Liker Chrome Extension is designed to automate the liking of tweets on the X platform (formerly Twitter). The extension interacts with the user's browser to perform specific tasks such as liking tweets, removing replies, and handling already liked tweets. The main components of the extension include the background script, popup script, and content script.

Main Components and Their Functions

1. **Background Script (background.js):**
    a. Initialization: The background script is triggered when the extension icon is clicked. It checks if the current tab is on x.com and opens the popup if it is.
    b. Tab Management: Stores the current tab ID in local storage and manages popup windows.
    c. Messaging: Handles messages from other scripts, such as updating the badge text and forwarding user messages.

2. **Popup Script (popup.js):**
    d. UI Interaction: Manages the user interface elements in the popup, such as the start/stop buttons, input fields, and progress bar.
    e. User Input: Captures the number of likes the user wants to perform and starts the liking process.
    f. Messaging: Listens for messages from the background script and updates the UI accordingly.

3. **Content Script (contentScript.js):**
    g. Tweet Interaction: Contains the main logic for interacting with tweets, including liking tweets, removing replies, and handling already liked tweets.
    h. Automation: Uses asynchronous functions to perform tasks with delays, simulating user behavior to avoid detection.
    i. Messaging: Sends status updates and results back to the popup and background scripts.

Flow of the Extension

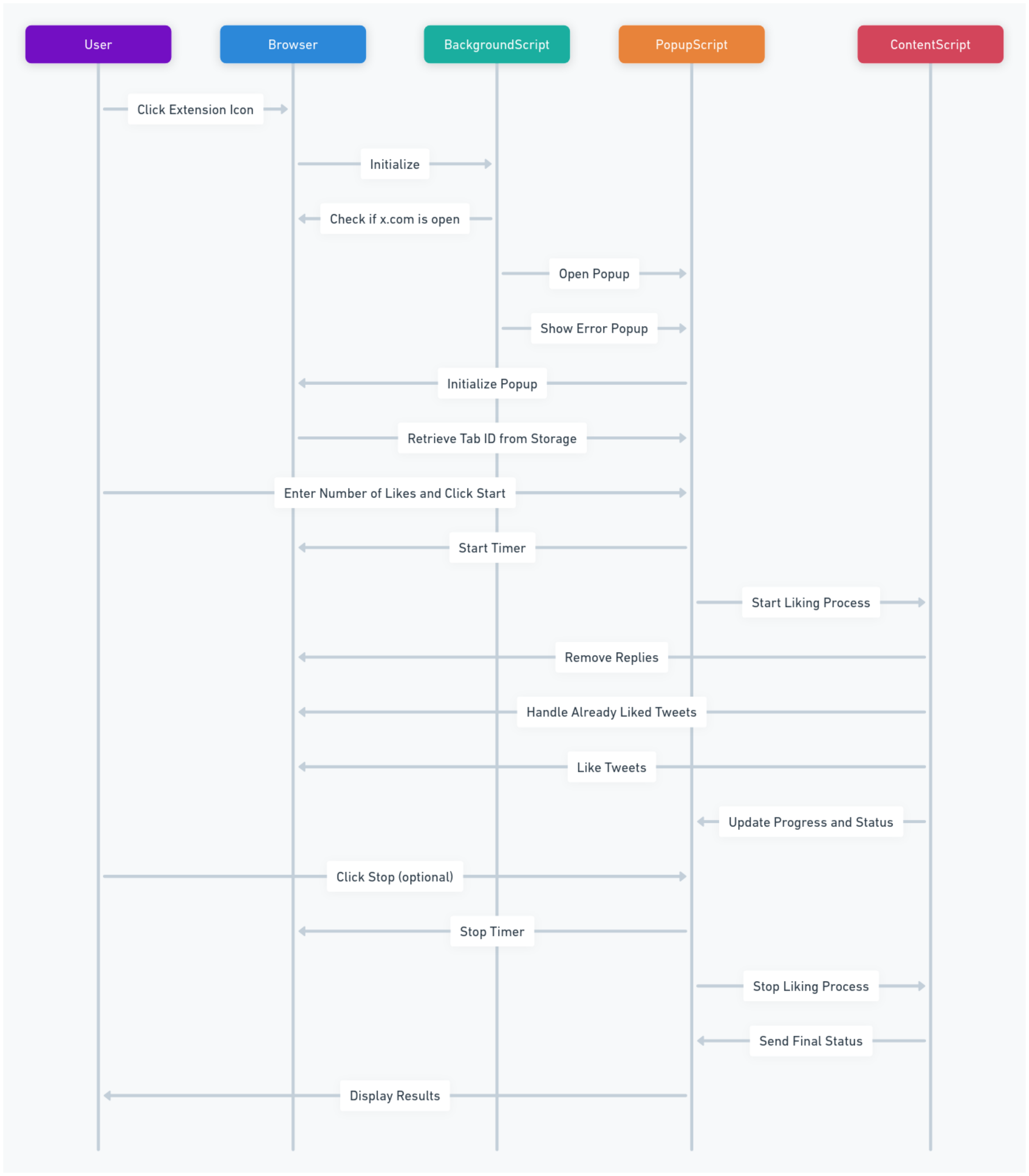
1. **User Interaction:**
    a. The user clicks on the extension icon.
    b. The background script checks if the current tab is on x.com.
    c. If true, it opens the popup window; otherwise, it shows an error popup.

2. **Popup Initialization:**
    a. The popup script initializes and retrieves the current tab ID from local storage.
    b. It sets up event listeners for the start and stop buttons.

3. **Starting the Liking Process:**
    a. The user enters the number of likes in the input field and clicks the start button.
    b. The popup script starts the timer and sends a message to the content script to begin the liking process.

4. **Content Script Execution:**
    a. The content script begins by removing replies and handling already liked tweets.
    b. It tracks SVG elements representing the like buttons and clicks them.
    c. It sends updates back to the popup script, which updates the progress bar and status messages.

5. **Completion and Stopping:**
    a. The content script continues liking tweets until the specified number is reached.
    b. The user can stop the process at any time by clicking the stop button.
    c. Once completed, the popup script resets the UI elements and stops the timer.


User
Browser
BackgroundScript
PopupScript
ContentScript
Click Extension Icon
Initialize
Check if x.com is open
Open Popup
Show Error Popup
Initialize Popup
Retrieve Tab ID from Storage
Enter Number of Likes and Click Start
Start Timer
Start Liking Process
Remove Replies
Handle Already Liked Tweets
Like Tweets
Update Progress and Status
Click Stop (optional)
Stop Timer
Stop Liking Process
Send Final Status
Display Results