Trailmix

Project Part 5: Implementation and Integration

Team 8

Sarah Cooper

Matthew Deagen

Leo Galang

Marc Ace Montesa

Instructors

David Feil-Seifer, Devrin Lee

External Advisors

Jay Thom, Nathan Thom, Maxwell Thom

Department of Computer Science & Engineering

University of Nevada, Reno

May 4, 2021

Table of Contents

Functional Requirements

The functional requirements completed are denoted with a C and the functional requirements not completed are denoted with an x and are explained in another table.

[1]	FR-01	Trailmix shall allow for a user to check their logged searches in the extension	C
[1]	FR-02	Trailmix shall not hold any information regarding the users that does not pertain to Trailmix.	C
[1]	FR-03	**Trailmix shall update all profiles everyday at 5:00 PST with a new set of randomized preferences	C
[1]	FR-04	Trailmix shall have a pop down user interface in the browser	C
[1]	FR-05	Trailmix shall include a list of preferences that are currently set for the user	C
[1]	FR-06	Trailmix shall include a way to remove preferences from a profile	x
[1]	FR-07	Trailmix shall include a reload of the page to enact any changes	C
[1]	FR-08	**Trailmix shall use an API to collect preferences of the users	C
[1]	FR-09	Trailmix shall allow for the user to pause the extension	C
[1]	FR-10	Trailmix shall allow for the user to resume the extension	C
[1]	FR-11	Trailmix shall allow users to change their user settings	C
[1]	FR-12	Trailmix shall allow users to generate a new profile	C
[1]	FR-13	Trailmix shall require users to create an account	C
[1]	FR-14	Trailmix shall require users to use a 10 character long password	C
[1]	FR-15	Trailmix shall require users to create a username without their real name	C
[2]	FR-16	Trailmix shall wipe all logs at 5:00 PST after FR-13 has been enacted	x
[2]	FR-17	Trailmix shall not show other users, even their fake profiles	C

[2]	FR-18	**Trailmix shall only collect the searched topics of the users in non-private mode	C
[2]	FR-19	Trailmix will not add certain searches to the preference list based on a blacklist of words	x
[3]	FR-20	Trailmix shall show a list of websites that are not being logged	x
[3]	FR-21	Trailmix will allow users to toggle on or off the list of not-logged preferences	x
[3]	FR-22	Trailmix will include a list of okay domains which will be logged	x
[3]	FR-23	Trailmix shall include a list of preferences that the user has already accepted	x
[3]	FR-24	Trailmix shall allow users to include preferences	x

Non-Functional Requirements

NR-01	Trailmix will work on Windows 7+/MacOS/Linux	C
NR-02	Trailmix will work on Android/ iOS	x
NR-03	Trailmix will require Chrome browser	C
NR-04	Trailmix shall maintain a simple user interface	C
NR-05	Trailmix shall maintain an easy to understand interface	C
NR-06	**Trailmix shall be implemented with JavaScript (React), HTML and CSS	C
NR-07	Trailmix shall not be open source	C
NR-08	Trailmix should never collect user information including: name, ip address, etc.	C
NR-09	Trailmix shall maintain user privacy and security	C
NR-10	Trailmix shall only take up the minimum amount of computer resources	C

** Note: Team 8 made revisions on the wording of these functional requirements that does not take away from the intended functionality.

Errata:

Level	Requirement ID	Old Wording	Revised Wording
[1]	FR-03	**Trailmix shall update all profiles everyday at 5:00 PST with a new set of randomized preferences	Trailmix shall update a user's profile with a new set of randomized preferences **based on a chosen set time from settings.**
[1]	FR-08	**Trailmix shall use an API to collect preferences of the users	Trailmix will use **chrome.history** to collect the search history of the users
[2]	FR-18	**Trailmix shall only collect the searched topics of the users in non-private mode	Trailmix does not automatically collect searched topics in non-private mode. **The user enables this setting through Chrome's extension settings.** The Trailmix team does not condone this.
	NR-06	**Trailmix shall be implemented with JavaScript (React), HTML and CSS	Trailmix shall be implemented with **JavaScript**, HTML, and CSS

Functionality Not Implemented

		Functional Requirement	Reason it was not completed
[1]	FR-06	Trailmix shall include a way to remove preferences from a profile	Not completed because we realized that the users can just generate new profiles and therefore removing one preference did not make sense when the user should change the whole profile
[2]	FR-16	Trailmix shall wipe all logs at 5:00 PST after FR-13 has been enacted	Because we update the logs with the user's history which can also include the searches made by Trailmix, we opted to not wipe the logs.
[2]	FR-19	Trailmix will not add certain searches to the preference list based on a blacklist of words	Decided not to implement a white or black list. This should have been an FR 3 with the other black and white list requirements.

Contributions of Team Members

Sarah Cooper worked with Matthew and Marc with the UI, CSS, and JavaScript of Trailmix. She wrote about 695 lines of code throughout the popup.html, trailmix.css, popup.js and background.js; setting up the framework for the code. This includes but is not limited to: the HTML file, not including the settings and creating page containers; the background.js except for some of the message request conditional statements that were done by Marc; and the popup.js minus a few functions which were done by Marc and Matthew. The functions done by Marc and Matthew are designated in the code as well in their contributions. She also set up the database and got the code working with Firebase. Originally Sarah tried to get MySQL to work, however that did not work and the team moved on to using Firebase. The amount of time Sarah spent working on the code, including spending a lot of time trying to get MySQL to work, was about 3.5 weeks.

Matthew Deagen helped Sarah with the functionality that sends the user's browser history to the Trailmix server. He also worked on the functionality for the refresh button. This function would take random preference entries from the server, open each of them in a new tab, then close all of the new tabs automatically. Earlier in the project, Matthew helped with the UI of the extension such as making spacing corrections with CSS rules, and also created the UI for the create account page and the settings page. Matthew also made the extension icon svg/png's. The amount of code written was about 255 lines of code. All of this was around 23 hours of work.

Leo Galang worked on the prototype and UI (what buttons/functionality were included in the extension and also how it looked -- color scheme, font, button placement). He also worked on the script for the video and presentation, and project documentation for project assignment 5. He spent around 9 hours.

Marc Ace Montesa did the auto-refresh timer functionality: from retrieving the preferred refresh time from users, to setting that time on the profiles; retrieving and displaying the current trail of a given user into the extension's user interface; as well as posting the history of the user. In terms of the databasing, Marc helped work on the database, as well as presenting the idea of using Firebase compared to the original plan of using MySQL, SQLite, or Node.js as previously stated. He also worked on the project video: including filming and editing. The video took about 6 hours all-in-all, the coding took roughly a week, and he coded about 200 lines of code.

Software Overview

Name	Document / Line	Description
background.js	Document	Backbone of program. Background functionality (generating profiles, refreshing user details). Communicates between code and Firebase database.
firebase.initializeApp()	Line 17 -> background.js	Initializes database to be used throughout the code.
chrome.runtime.onMessage.addListener()	Line 28 -> background.js	How background.js communicates with popup.js. All of the following if statements are found within this function.
if (request.type == “credentials”)	Line 32 -> background.js	Checks to see if messages received from popup.js are credentials, then executes functionality.
if (request.type == “set_refresh_time”)	Line 67 -> background.js	Resets refresh time for the user.
if (request.type == “dont_send”)	Line 82 -> background.js	If a user doesn’t opt into sending their data to Trailmix.
if (request.type == “send”)	Line 91 -> background.js	If a user wants to send their data to Trailmix.
if (request.type == “pref_time”)	Line 102 -> background.js	Checks time that the user last refreshed.
if (request.type == “check_state”)	Line 133 -> background.js	Checks if the user is logged in.
if (request.type == “auto_refresh”)	Line 143 -> background.js	Updates refresh time for the user.
if (request.type == “function_change”)	Line 182 -> background.js	Enable/disable functionality.
if (request.type == “profile”)	Line 201 -> background.js	Retrieves/updates user profiles.

if (request.type == “signup”)	Line 239 -> background.js	New account generation.
if (request.type == “post_history”)	Line 284 -> background.js	Updates user history.
function generate_new_profile()	Line 338 -> background.js	New profile generation.
popup.js	Document	All event listeners that communicate between HTML and background.js.
chrome.runtime.onMessage. addListener()	Line 42 -> popup.js	Communicates between popup.js and background.js. All of the following if statements are found within this function.
if (request.type == “auth”)	Line 44 -> popup.js	Checks user credentials.
if (request.type == “new_profile”)	Line 68 -> popup.js	Requests new profile generation.
if (request.type == “user_refresh”)	Line 72 -> popup.js	Communicates user refresh details.
if (request.type == “update_logs”)	Line 75 -> popup.js	Communicates with update_logs.
if (request.type == “curr_profile”)	Line 79 -> popup.js	Retrieves current profile.
if (request.type == “state_checked”)	Line 86 -> popup.js	Checks to see if the user is logged in.
if (request.type == “dont_send_successful”)	Line 98 -> popup.js	Checks if the user successfully opted out of sending data to Trailmix.
x.addEventListener	Lines 112-324 -> popup.js	Describes state and functionality of all buttons.
function updatelog(logs_file)	Line 327 -> popup.js	Updates log file.
function updatelist(profile_display)	Line 341 -> popup.js	Updates profile list.
function run_matthews_code	Line 384 -> popup.js	Pulls random preferences from the server and automatically creates a

		new tab for each preference. Also deletes these tabs.
function SHA256()	Line 418 -> popup.js	Hashing function for sensitive data.
manifest.json	Document	Manifest file needed for Chrome extensions.
package.json	Document	File necessary for node.js (not used in code).
package_lock.json	Document	File necessary for node.js (not used in code).
popup.html	Document	How the extension is displayed to the user, and how the functionalities are provided as well.
trailmix.css	Document	Styling; how the extension appears (positioning, color, etc).

Source Code

*Attached as a zip file

To run the code:

1) Extract the zip file into a folder of your choice.
2) Open Chrome and go to “chrome://extensions”
3) Click on the “Developer mode” in the upper-right corner
4) Then choose “Load unpack” and choose the Trailmix folder
5) It should show up in the extension dropdown
6) GOOD TO GO!

If you need extra help: how to install a chrome extension with images

If you have any questions about running the code, please let someone in Team 8 know, and we can provide assistance.