МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Комп'ютерних наук,
управління та адміністрування
Кафедра Інформаційних технологій

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: Розробка плагіну для браузера з функціями захвату медіаконтенту

Виконав студент 4 курсу групи К-25
Спеціальність 122 Комп'ютерні науки
Федченко Максим Сергійович

Керівник ст.викл.
Рольщиков Вадим Борисович

Консультант к.геогр.н., доцент
Кузніченко Світлана Дмитрівна

Рецензент к.геогр.н., доцент
Лужбін Анатолій Михайлович

Одеса 2020

ЗМІСТ

ВСТУП

Функція здійснення відео захвату екрану використовується вже давно. Останнім часом частіше використовується термін скринкастинг, тобто цифрова аудіо та відеозапис, яка проводиться безпосередньо з монітора комп'ютера.

Як правило, цей інструмент використовується для ширшого доступу та динамічного подання того чи іншого цифрового додатки. Однак те, що відбувається на екрані може стосуватися і зовсім інших речей.

Фахівці підкреслюють, що особливістю скринкастингу є можливість задіяти одразу кілька «каналів сприйняття інформації»: зоровий, моторний і слуховий.

Скринкастинг містить в собі безліч способів його застосування, більшість яких пов'язані з процесом здобуття освіти, перш за все, з дистанційним навчанням:

- огляд програмного забезпечення, електронного навчального посібника, будь-якого цифрового освітнього ресурсу;
- демонстрація послідовності дій з програмою, додатком, іншими словами своєрідна відео інструкція про алгоритм дій;
- запис під час проводження тестів над програмним продуктом; для більш легкого відстеження помилок, розробникам може знадобиться запис процесу тестування;
- інтерактивна дошка вчителя; вчитель використовує екран комп'ютера для трансляції одному або декільком учням відео уроку, в якому він грає активну роль своїми коментарями, підкресленням, демонстраціями медіа файлів.

Таким чином оскільки функція запису екрану має велику кількість застосувань та є незамінною для вирішення визначених завдань, вона є дуже актуальною для сучасного користувача.

Метою даного проекта є написання додатку з функціями запису медіа-пристроїв та екрану. Застосування повинно містити зручний та повний інтерфейс, з усіма необхідними для користувачей настройками.

Параметри записки:

- 21 посилання;
- 57 сторінок;
- 26 рисунків;
- 2 таблиці.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Моделювання процесу розробки

Даний проект у якості основних своїх вимог, які йому необхідно задовільнять, повинен містити функції: запису екрану, запису приймальних пристроїв звуку, запису відеокамери. Також мати можливість подальших модифікацій та не проблематичних усунень багів.

Розробка даного проекту складається з наступних кроків:

– аналізу предметної області;
– вибір засобів застосування;
– розробки програми;
– тестування програми.

Участь та зв’язок один з одним цих структурних кроків зображено на рис. 1, 2.

В результаті успішного виконання даних структурних частин маємо отримати програму, яка містить в собі всі перераховані функції і задовольняє поставлені перед нею вимоги. Були виявленні такі задачі для вирішення.

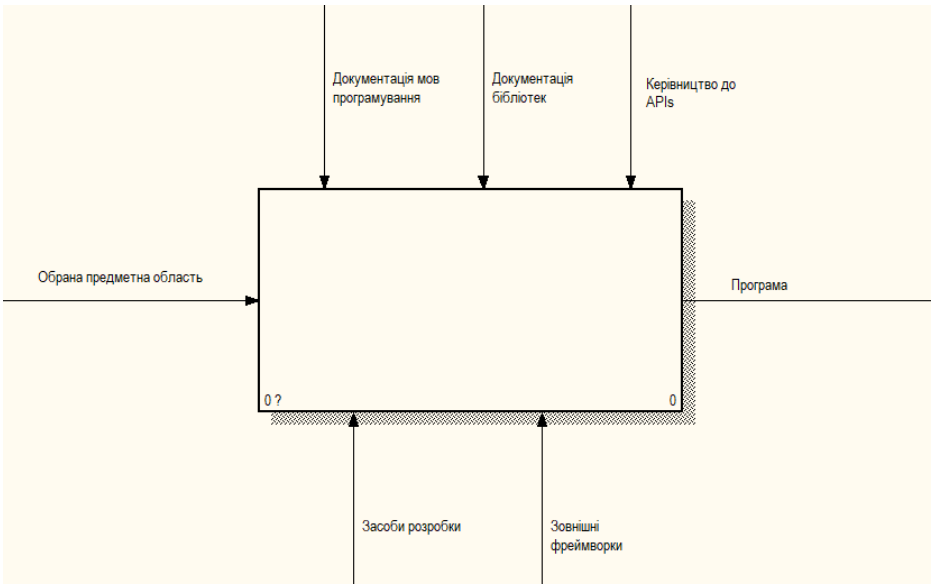

Рисунок 1 – Опис проекту в цілому

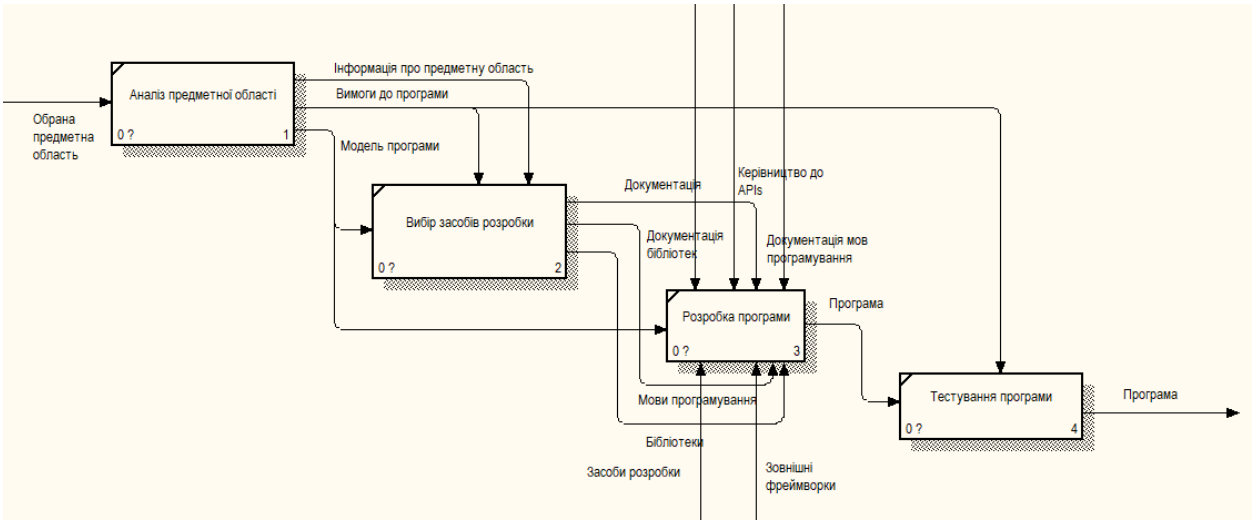

Рисунок 2 – Опис структурних частин розробки проекту

1.2 Опис предметної області

1.2.1 Опис технології скринкастингу

Скринкастингом можна вважати будь-який відеозапис, яка надходить безпосередньо з екрану комп’ютера.

Ця функція необхідна, коли потрібно показати те іншим людям те, що сам користувач бачить на моніторі комп'ютера. Іноді наявність простих нерухомих зображень є недостатнім, і якщо ілюстрація вимагає багато пояснень, скринкастинг може бути більш підходящим варіантом. Також скринкастинг може бути поєднаним із захопленням веб-камери, таким чином включаючи людський фактор.

Застосуванням, для якого скринкастинг зазвичай використовується та середньостатистичний користувач зустрічається, є контент виду практичних керівництв і демонстрацій. Технологія скринкастингу на початку була спрямована на розробку інтерактивних демонстрацій програмних продуктів і до сих пір вона застосовує в першу чергу для створення матеріалів щодо володіння комп'ютерними програмами. Але використання скринкастингу не обмежується тільки цією областю, скринкастинг можна з успіхом використову-

вати при розробці різноманітних навчальних матеріалів що відносяться до різних сфер знань. Дуже вдалим є використання цієї технології при наявності готових мультимедійних презентацій.

В галузі освіти, як зазначено у [1][1)] дана можливість застосування технології, в зв'язку з тим що одним із пріоритетних напрямків розвитку сучасної системи освіти є впровадження нових інформаційних технологій в освітній процес і використання інформаційних технологій, безсумнівно, відкривають додаткові можливості як для підвищення якості та ефективності процесу навчання, так і для розширення сфер його застосування, ставиться під пильну увагу фахівцями освітніх установ, які розглядають процес створення і впровадження електронних підручників, посібників, довідників, практикумів та інших навчальних матеріалів, розроблених на базі сучасних інформаційних технологій, частково або повністю здійсненим шляхом технології скринкастингу.

Технологія екранного відео є дуже вдалим рішенням в освітньому процесі: спостерігаючи за кожним рухами та словами, людина що навчається сама впроваджується в процес; може неодноразово прокручувати відео, загострюючи увагу на найбільш складних для нього моментах; є можливість вивчати матеріал в індивідуальному темпі; на практиці можна застосувати матеріал в режимі реального часу.

Така форма навчання дозволяє активізувати різні канали по одержанню інформації: і зоровий, і слуховий, і моторний. При їх комбінації відбувається інтенсифікація процесу навчання та значно зростає ступінь засвоєння матеріалу. Відомо, що при аудіо сприйнятті засвоюється тільки частина інформації, при візуальному значно більше а при їх комбінації як зазначається ЮНЕСКО, засвоюється до 65% отриманої інформації [2][2)]. Такий спосіб ви-

---

[1)] [1] How screen recording can be used in different ways [Sponsored] – Memeburn. URL: https://memeburn.com/2018/04/screen-recording-can-used-different-ways/ (дата звернення 22.05.2020)

[2)] [2] Developments in audio-visual education. URL: https://unesdoc.unesco.org/ark:/48223/pf0000001433 (дата звернення 22.05.2020)

користовуються викладачами, щоб зробити урок інтерактивним і захоплюючим.

Важливо, що навчальний матеріал, поданий за допомогою технології екранного відео, як правило, носить структурований, послідовний, цілісний та закінчений характер, оскільки його підготовка вимагає клопітливої попередньої підготовки автором. Крім того, електронні навчальні посібники, засновані на технології екранного відео, відрізняються інформаційною насиченістю та сильним емоційним впливом на учня.

Навчальні матеріали, створені за допомогою скринкастингу можна використовувати як самодостатні програмні продукти на аудиторних заняттях і при самостійній поза аудиторній роботі учнів, а також комбінувати їх з іншими засобами навчання і впроваджувати в цілі курси, наприклад в системах дистанційного навчання.

Крім освітніх і демонстраційних цілей дана технологія може мати і інші застування.

Інший корисний спосіб використовувати функції запису екрану – це запис і зберігання відео дзвінків або конференцій. На відміну від інших методів зв'язку, таких як електронна пошта або обмін миттєвими повідомленнями, на платформах відео дзвінків, як правило, відсутній будь-який метод збереження вмісту розмови.

Запис екрану може допомогти подолати цей пробіл, і це особливо корисно для важливих відео дзвінків або навіть ділових зустрічей, проведених у вигляді відеоконференцій.

Ще один спосіб пов'язаний з труднощами документування помилки або описом проблеми, з якою зіткнувся користувач при зверненні в службу технічної підтримки. Замість того, щоб робити спроби описати все словами, користувач може використовувати скринкастинг для запису відео про помилку або проблему, з якою він зіткнувся.

Хоча це, безумовно, один з найменш популярних способів використання екранної записи, такий принцип може заощадити користувачеві час.

І нарешті скринкастинг може використовуватися в сфері розваги. Є багато людей, яким подобається дивитися, як інші люди роблять різні речі, наприклад гри. Існує безліч відеохостингових сервісів, з яких одним з найбільш популярних є YouTube, які позволяють розміщувати на них відео, роблячи їх доступними для будь-якої людини з доступом до мережі Інтернет.

1.2.2 Опис програми плагін

На відміну від традиційних зображень або відео, записаних за допомогою камери, для захоплення зображення або записи дій на екрані необхідне відповідне програмне забезпечення.

Тип браузерного розширення у якості програмного забезпечення був обраний у зв'язку з малочисленими аналогами, поширеністю програми браузер, міжплатформеністю даного типу ПЗ, та також тенденцією користувачів все частіше замість програм на локальному комп'ютері використовують сервісні аналоги, які розташовані на серверах, і з якими користувач може працювати за допомогою браузера.

Для опису плагіна потрібно спочатку описати браузер. Браузер у житті сучасної людини однією з найважливіших комп'ютерних програм. Дозволяючи легким і зрозумілим для кожного користувача способом отримувати необхідну інформацію з мережі Internet, програма браузер стала одним з наймасовіших явищ.

Історія браузерів, є практично ровесниками Всесвітньої мережі, дозволяючи таким чином більше дізнатися і про сам Інтернет. Будучи формально лише звичайними комп'ютерними програмами, браузери фактично стали головним з'єднувальною ланкою між Інтернетом і людиною, і від того, як вони виконують покладені на них завдання, залежить і наше сприйняття віртуального світу.

Як вказано в [3][1] браузери містять в собі багато функцій. За допомогою браузера можна переглядати веб-сторінки, завантажувати файли, дивитися відео, слухати музику і переглядати документи. Все це дозволяє робити одна проста, але життєво необхідна програма.

Однак навіть такий функціонал, що надається даними програмами, може бути не достатнім для користувача. В силу уникнення накопичення непотрібних функцій, для браузерів, додатковий функціонал був винесений в окремі програми, які називаються розширеннями браузера, дозволяючи в разі потреби кожного індивідуального користувача доповнити спроможності браузера.

Сучасна концепція розвитку програмного забезпечення часто передбачає включення в основну програму тільки базових функцій, а додаткова функціональність реалізуються за допомогою доповнень. Браузери є типовими представниками такої ідеології.

Оскільки браузери зараз є дуже важливими програмами для людини, і є в будь-якому комп'ютері, а часом і кілька відразу, то їх творці надають необхідну інформацію для написання плагінів будь-яким бажаючим, тому плагінів було створено величезна кількість на будь-який випадок. Є дуже популярні плагіни, а є специфічні затребувані тільки вузькою групою осіб. Розширення дозволяють додавати в браузер тільки потрібні можливості, уникаючи накопичення функцій, які не використовуються.

Таким чином розширення для браузера є комп’ютерною програмою, яка розширює функціонал браузера. Також їх називають plug-in (плагін) або add-on (доповн5ення), і зараз вони підтримуються практично всіма браузерами. Найпопулярніші розширеннями розробляються для браузерів Google Chrome [4][2] і Firefox.

---

[1] [3] What's an Internet Browser? A Layman's Guide. URL: https://www.thebalanceeveryday.com/what-is-internet-browser-892819 (дата звернення 22.05.2020)

[2] [4] Develop Extensions – Google Chrome. URL: https://developer.chrome.com/extensions/devguide (дата звернення 22.05.2020)

Можливостями розширення є:

- зміна елементів управління браузера;
- перетворення завантажуваних сторінок шляхом додавання своїх скриптів і файлів стилів;
- створення фонових скриптів;
- задавання гарячих клавіш;
- додавання опцій в контекстне меню;
- створення сторінки опцій, для настройки параметрів розширення;
- використання зовнішньої програми, яка буде взаємодіяти з розширенням;
- використання специфічних для конкретного браузера функцій.

Залежно від мети з якою створюється розширення існує багато видів розширень, з яких можна виділити наступні:

- інтеграційні розширення; Такий тип розширення відрізняється тим що істотний свій функціонал розміщує на сервері, на якому розташовується хмарний сервіс, таким чином за допомогою функції експорту даних, налагодити процес обміну даними між різними внутрішніми системами.
- розширення сервіси; це мікро додатки, які викликаються по кнопці та взаємодіють зі сторінкою; потрібні вони для сервісів, які має сенс запускати в середовищі іншого сайту / додатка; він здатний зчитувати інформацію з сайту, на якому його викликали, і тим самим істотно спростити користувачеві взаємодію; це можуть бути будь-які сервіси заміток, скріншотів або відео, вони можуть не просто зберігати дані, а відразу вивантажувати в хмару, або відправляти, наприклад поштою.
- також можна виділити розширення інструментів розробника, які полегшують розробку сайтів або веб-додатків.

Переваги плагінів, як вказано у [5][1]:

- швидкість доступу, зручність і зрозумілість використання;
- міжплатформеність – здатність працювати на будь-якій платформі, де є браузер;
- можливість інтегрувати неінтегроване, тобто вставити свою функціональність в сторонні продукти, в ядро яких доступу немає;
- можливість об'єднувати свої системи і хмарні сервіси в комплексний корпоративний ландшафт систем.

Недоліки браузерних плагінів [5][1]:

- необхідність періодичних оновлень плагіна під оновлення браузера або сервісу;
- під кожен браузер потрібно писати окрему версію розширення.

1.3 Аналіз існуючих аналогів

1.3.1 Аналіз додатку Nimbus

Одним з аналогів для розгляду є додаток Nimbus. Nimbus [6][2] – це спеціальний плагін, який служить для запису відео та скріншотів з подальшим їх редагування.

На рис. 3 зображено головне меню Nimbus з основними функціями, яке викликається при натисканні на кнопку розширення Nimbus.

При виборі функції запису екрану з'являється інше вікно з налаштуваннями додатку (див. рис. 4).

Також, як зображено на рис. 5, є можливість відкриття спеціальної вкладки для більш докладних налаштувань процесу запису.

---

[1] [5] Что такое расширения браузера? Их виды и польза. URL: https://evergreens.com.ua/ru/articles/browser-extensions.html (дата звернення 22.05.2020).

[2] [6] Nimbus Note - Самое удобное приложение для создания, редактирования и хранения заметок. URL: https://nimbusweb.me/ (дата звернення 22.05.2020)

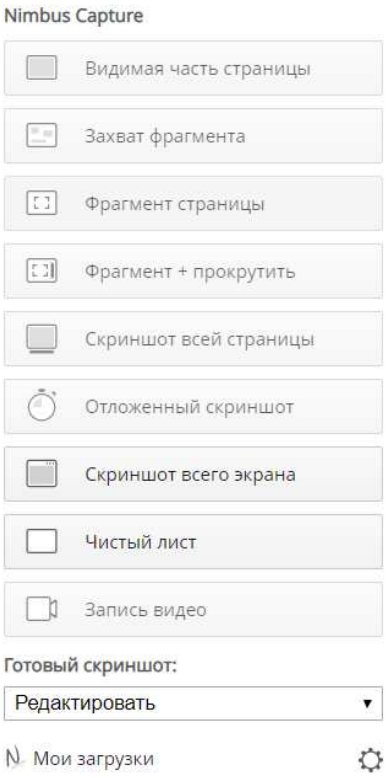

Рисунок 3 – Основні функції плагіна Nimbus

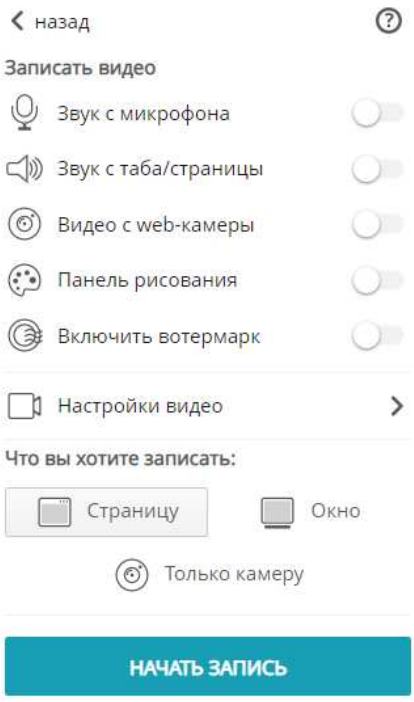

Рисунок 4 – Вікно з налаштуванням процесу запису

За даними зображень видно що Nimbus має достатню для більшості користувачів кількість опцій по запису екрану і камери.

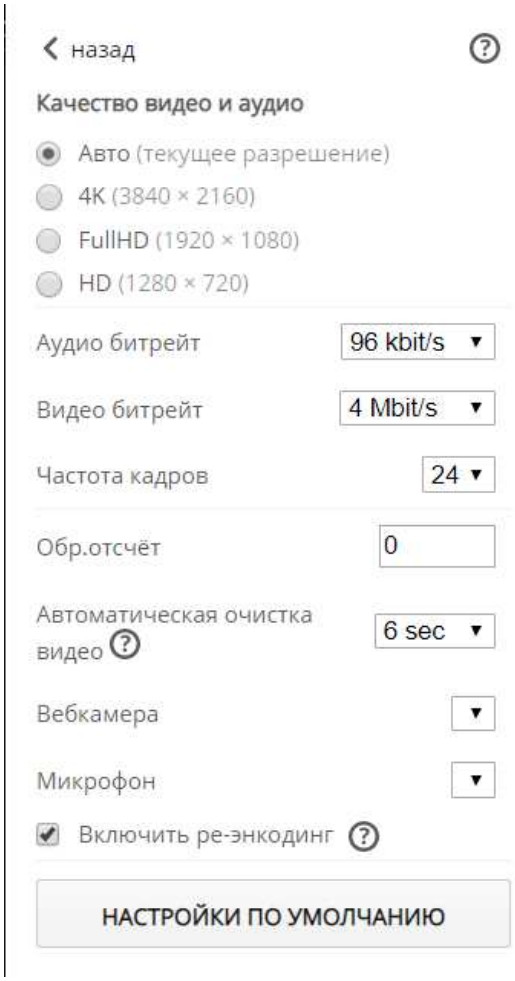

Рисунок 5 – Налаштування функцій додатку

Після запису відео дозволяє користувачеві функції з експорту файлу (див. рис. 6) на локальну машину або на хмарний сервіс Google drive або YouTube (для платної версії).

Крім цього, як видно на рис 7, плагін надає додаткові можливості по редагуванню відео.

Після аналізу додатку Nimbus були виявлені наступні недоліки:

– наявність надлишкових, для конкретних потреб користувачів з відеозапису, функцій для роботи з зображеннями.
– обмеження з експорту відеозаписів на хмарні сервіси, регламентовані платною версією.
– нагромаджених інтерфейс.

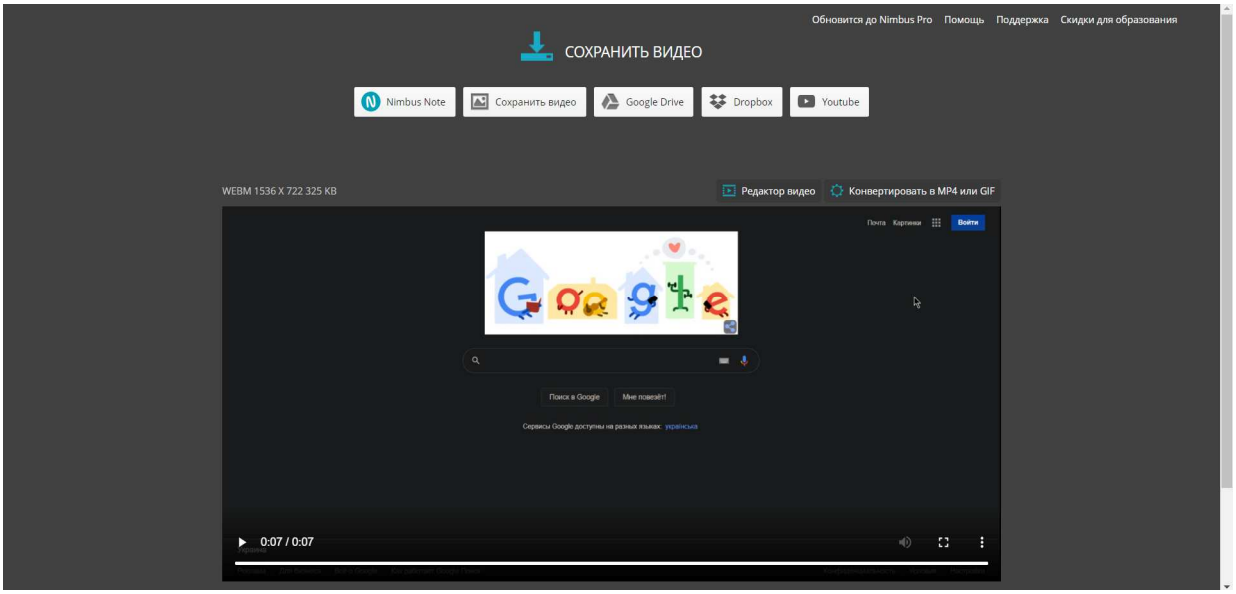

Рисунок 6 – Web-сторінка з опціями з експорту відео

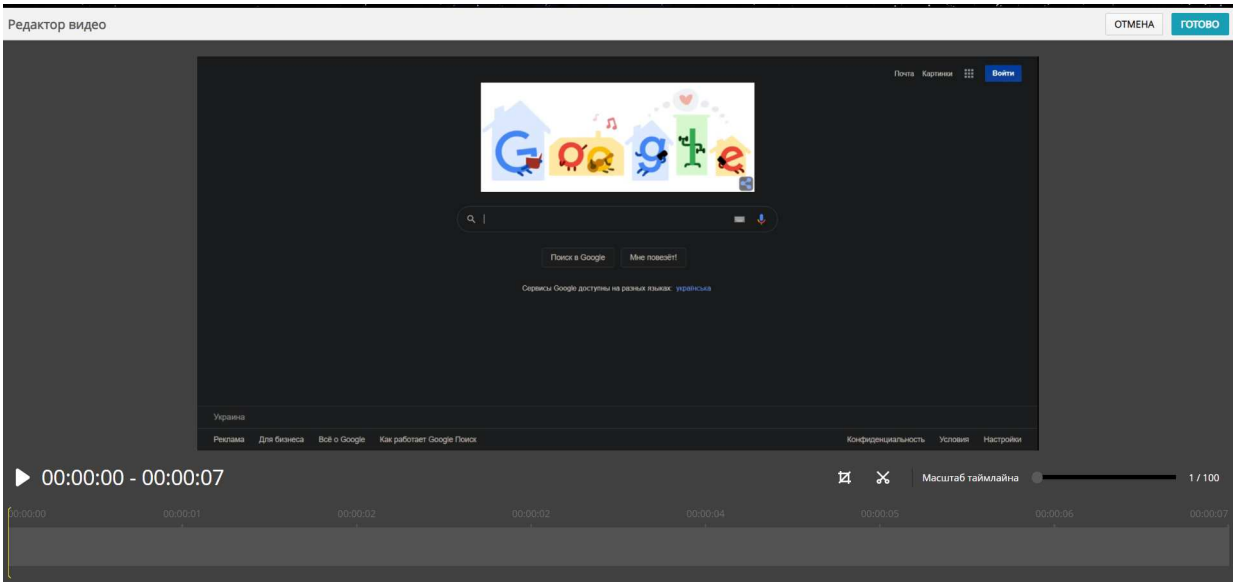

Рисунок 7 – Web-сторінка для редагування відео

1.3.2 Аналіз застосування Screencastify

Наступним розглянутим додатком є плагін Screencastify [7][1].

Screencastify – являє собою плагін для запису, редагування і розміщення на хмарних сервісах відеофайлів.

При натисканні на кнопку розширення відкривається вікно (див. рис. 8), яке представляє собою головне меню додатка:

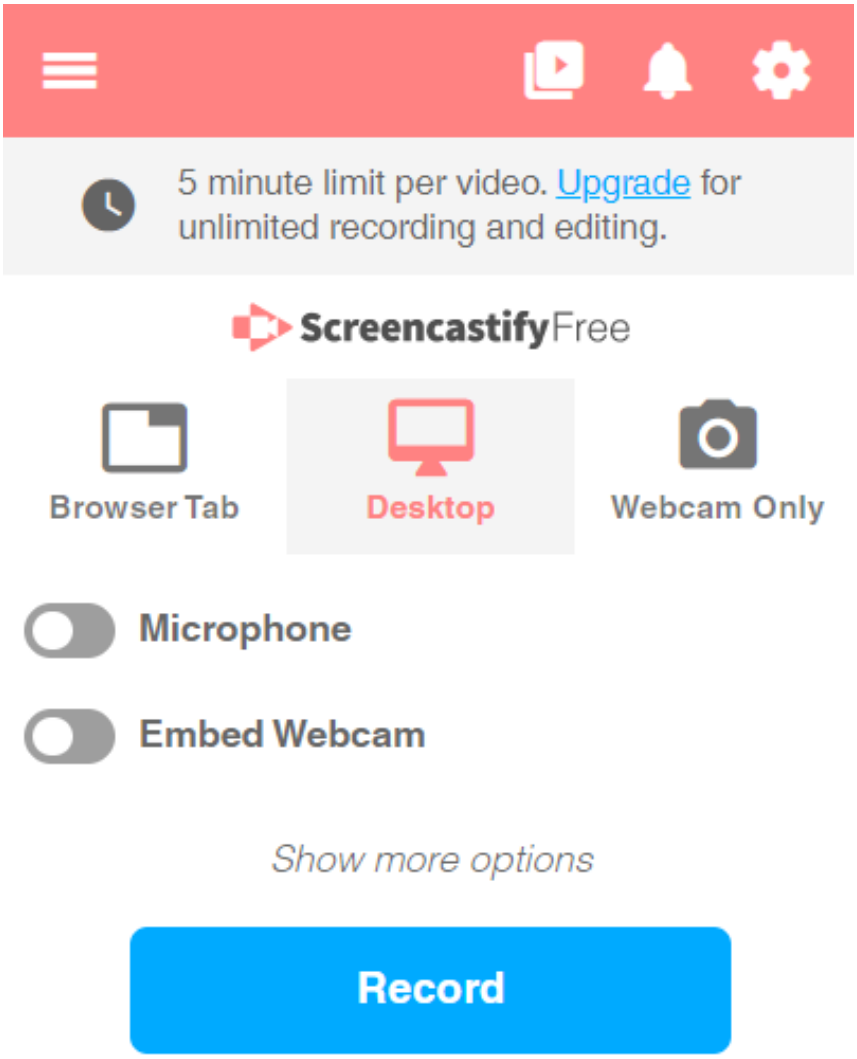

Рисунок 8 – Головне вікно додатку Screencastify

[1] [7] Screencastify | The #1 Screen Recorder for Chrome/URL: https://www.screencastify.com/ (дата звернення 22.05.2020)

Корисною стороною розширення Screencastify є те що він має зручний інтерфейс відображає необхідні функції для задоволення потреб користувачів по запису і експортуванню відеоматеріалів.

Після запису відео дозволяє користувачеві функції з експорту файлу на локальну машину, на хмарний сервіс Google drive або YouTube (для платної версії) або відправити за допомогою пошти Gmail (див. рис. 9).

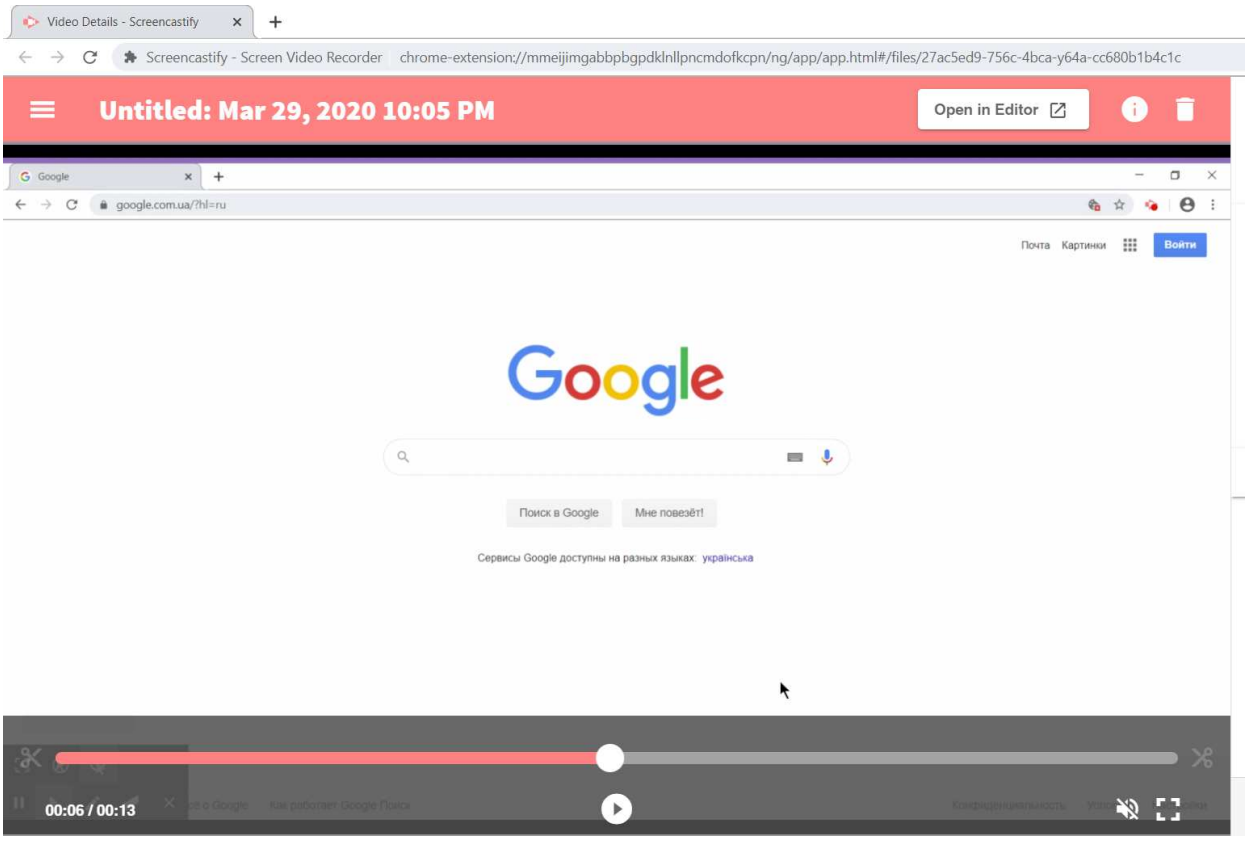

Рисунок 9 – Web-сторінка для редагування та експортування відео

Також, як можна побачити на рис. 10, за допомогою відповідної опції можна переглянути всі зроблені записи, що зберігаються локально або на хмарному сервісі GoogleDrive.

Крім того розширення має групу високорівневих налаштувань з регулювання процесу запису та гарячих клавіш (див. рис.11).

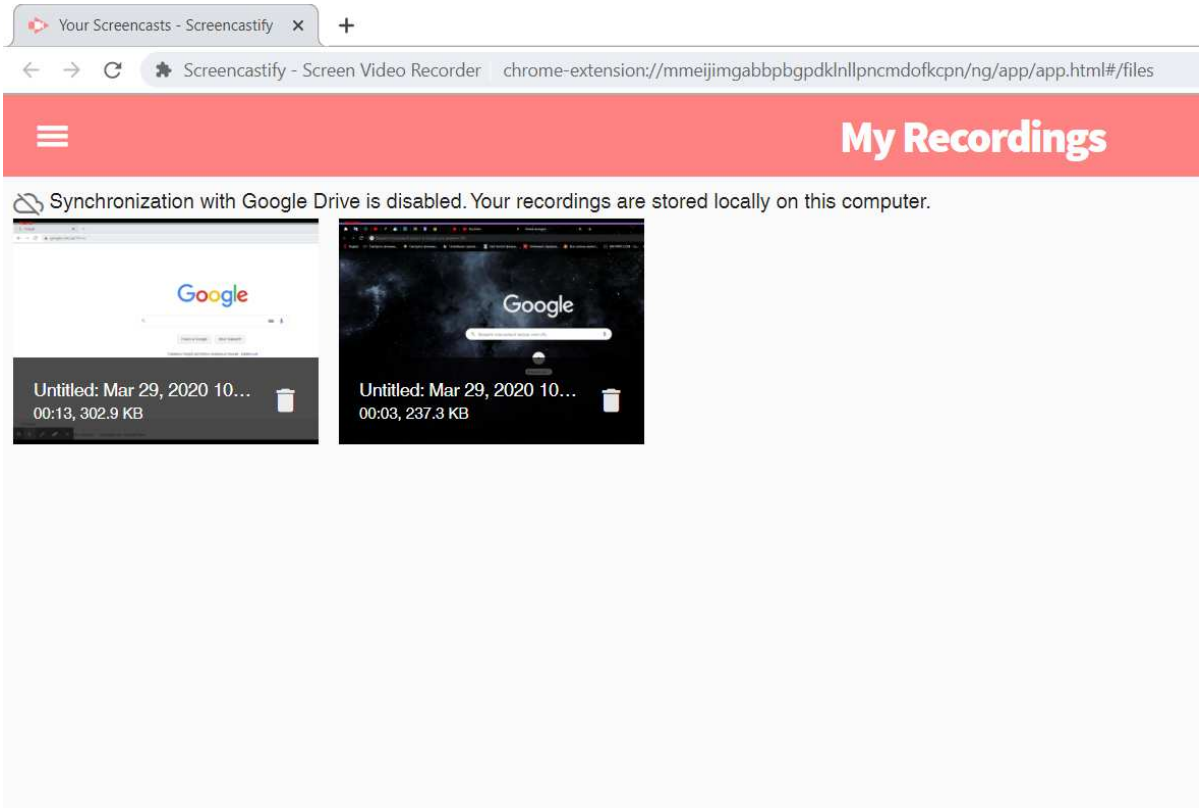

Рисунок 10 – Web-сторінка для перегляду зроблених записів

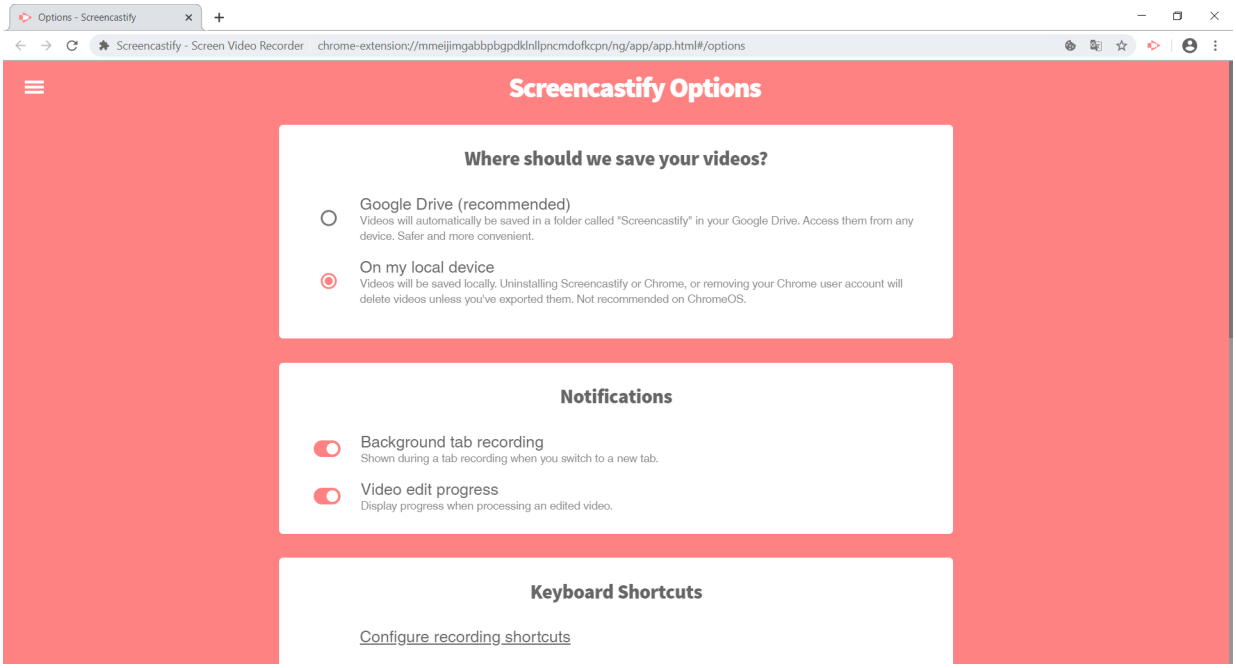

Рисунок 11 – Налаштування плагіну

Після аналізу додатку Screencastify були виявлені наступні недоліки:

- обов'язковість авторизації google account для надання доступу до можливостей розширення.
- суворі функціональні обмеження безкоштовної версії програми, зокрема обмеження часу запису.
- обмеження в докладних налаштуваннях запису відеоматеріалу.

1.4 Характеристика об'єкта розробки

З урахуванням недоліків існуючих аналогів плагінів по запису і роботи з відео в браузері, які перераховані в підрозділі 3, були складені вимоги до готового продукту плагіна:

- плагін повинен мати функцію запису екрану, а також конкретного вікна або вкладки браузера, на яких буде відображатися робота користувача, з подальшим збереженням відеозапису, в визначений користувачем каталог;
- плагін повинен мати функції роботи з відеокамерою, також ці функції повинні бути узгоджені з записом екрану;
- плагін повинен мати функції роботи з мікрофоном, також ці функції повинні бути узгоджені з записом екрану;
- для відповідного і зручного режиму запису відеоматеріалів, плагін повинен мати всі необхідні настройки, для можливості користувача налаштувати функції плагіна згідно зі своїми вимогами до його використання; але в плагіні не повинна бути присутня надмірність функцій, не відповідних до специфіки і спрямованості цього додатка;
- плагін повинен мати функцію вибору місця в локальному сховищі за допомогою провідника для збереження записаних медіа-файлів.

1.5 Проектування структури програми

Для спрощення розробки даної системи були побудовані UML діаграма класів, яку можна побачити на рис. 12, та діаграма послідовності, яку можна побачити на рис. 13. Побудова діаграм здійснювалась згідно з особливостями розробки браузерних плагінів [4][1].

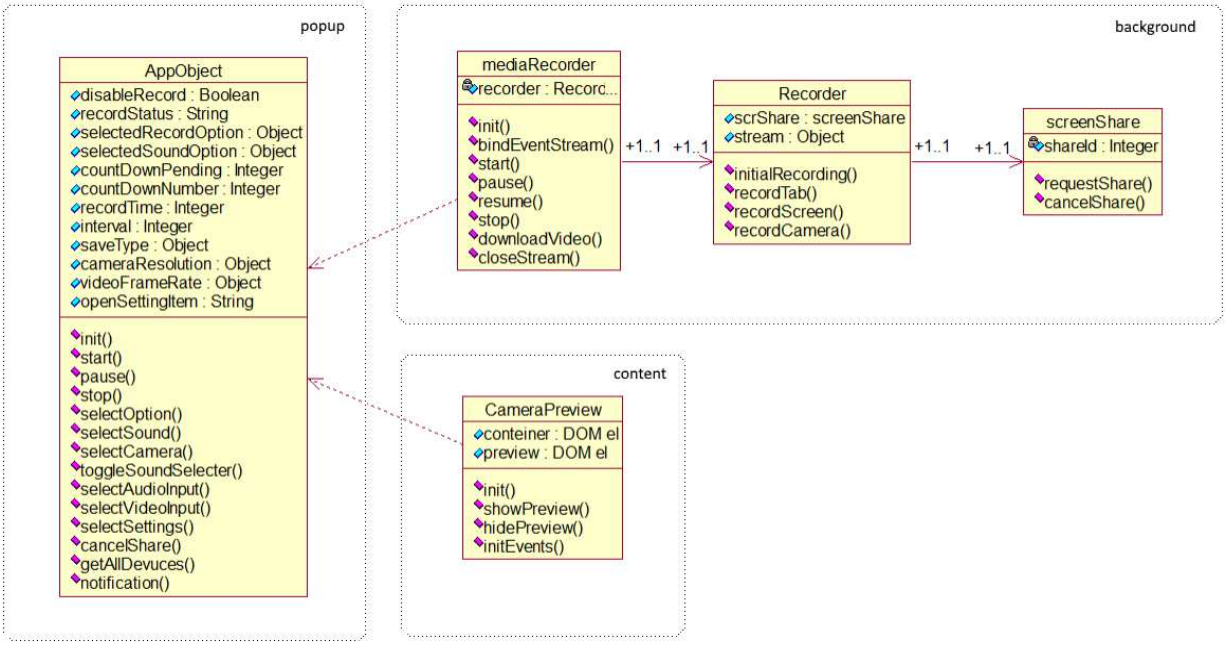

Рисунок 12 – UML діаграма класів

Побудовані UML діаграми описують структуру додатку.

Діаграма класів зображає взаємозв'язок між класами плагіну. Також помічаються поля та методи класів.

Діаграма послідовності зображає часову послідовність передачі повідомлень між компонентами браузерного плагіну.

[1] [4] Develop Extensions – Google Chrome. URL: https://developer.chrome.com/extensions/devguide (дата звернення 22.05.2020)

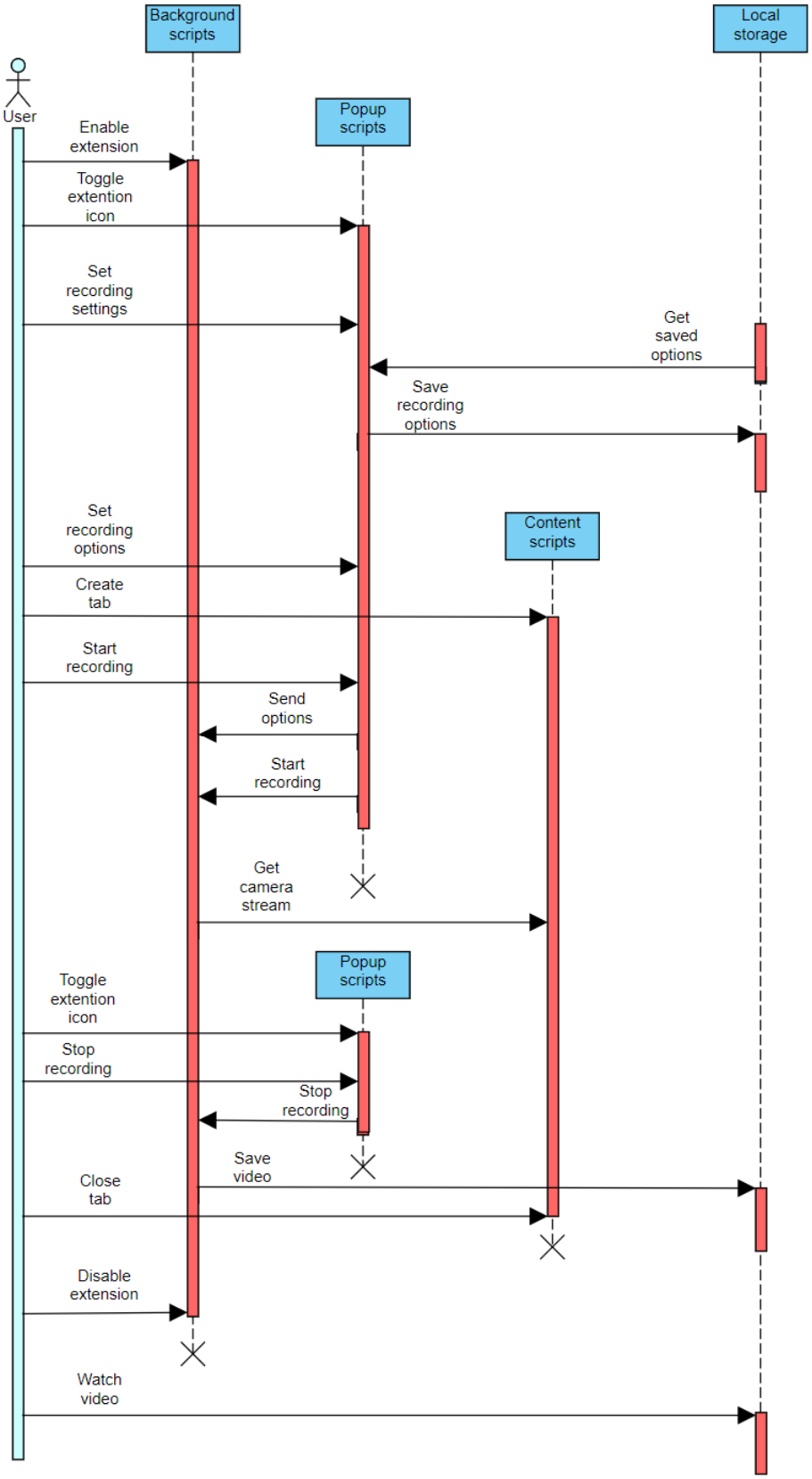

Рисунок 13 – UML діаграма послідовності дій додатку

2 ВИБІР ЗАСОБІВ РОЗРОБКИ

В даному розділі детально будуть обрані мови написання даного додатку, середовище його написання а також додаткові застосування необхідні для його здійснення.

Структура розділу «Вибір засобів застосування» та зв’язок його етапі зображено на рис. 14.

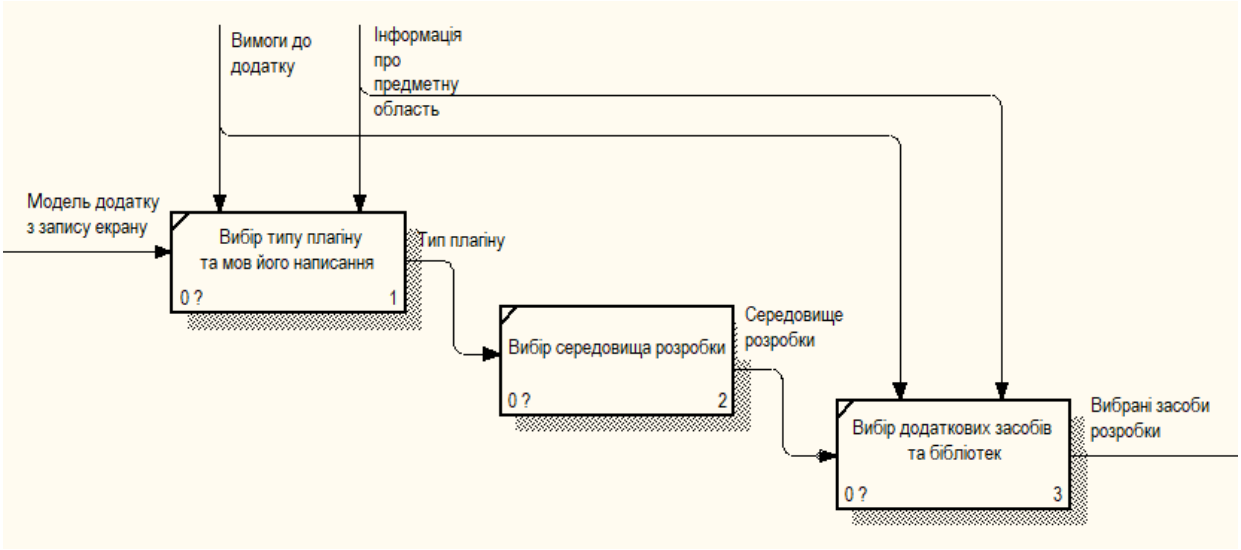

Рисунок 14 – Опис структурних елементів розділу

2.1 Вибір типу плагіну та мов його написання

Для вибору типу плагіна і відповідно його засобів, якими він буде розроблятися необхідно було вибрати браузер на чию технологію плагін і буде орієнтуватися розробляємий плагін і на базі якого буде функціонувати.

Для цього, згідно ресурсу [8][1)], було розглянуто список популярних для користувачів браузерних додатків.

[1)] [8] Статистика самых популярных браузеров в мире – Marketer https://marketer.ua/stats-of-browsers-2017/ (дата звернення 22.05.2020)

Список поширених браузерів:

– «Chrome»;
– «Opera»;
– «Firefox»;
– «Safari»;
– «Yandex Browser»;
– «Internet Explorer»;
– «Microsoft Edge».

Також, як видно на діаграмі, зображеної на рис. 15, була досліджена статистика процентного співвідношення до використання браузерів користувачами у світі і в відповідності з діаграмою на рис. 16 в Україні.

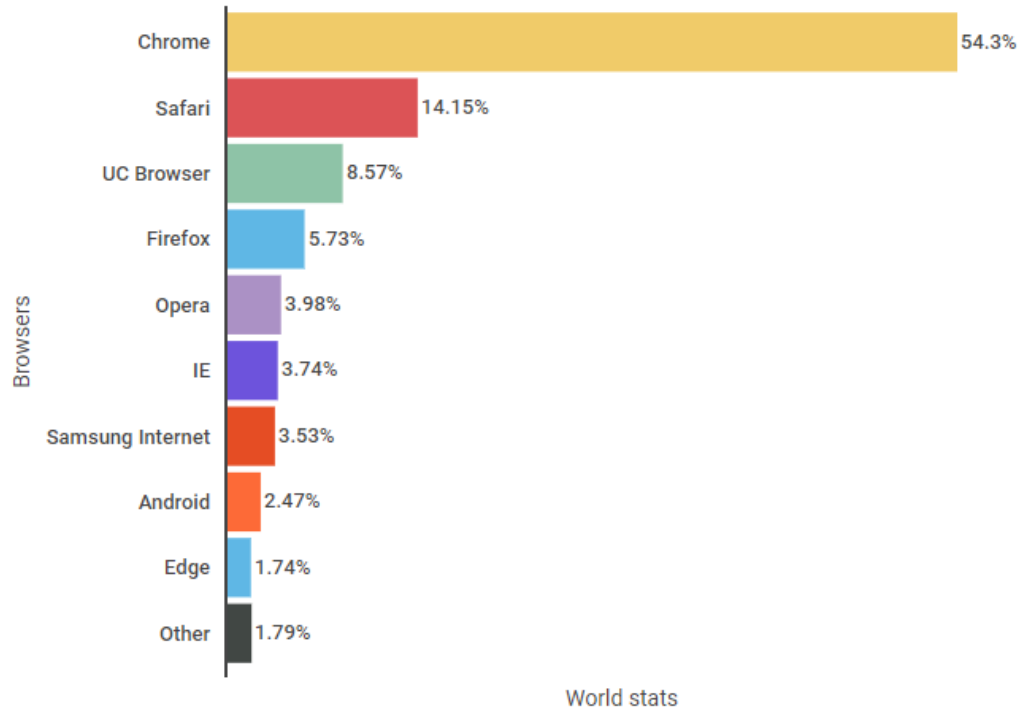

Рисунок 15 – Статистика з використання браузерів у світі [8][1)]

[1)] [8] Статистика самых популярных браузеров в мире – Marketer https://marketer.ua/stats-of-browsers-2017/ (дата звернення 22.05.2020)

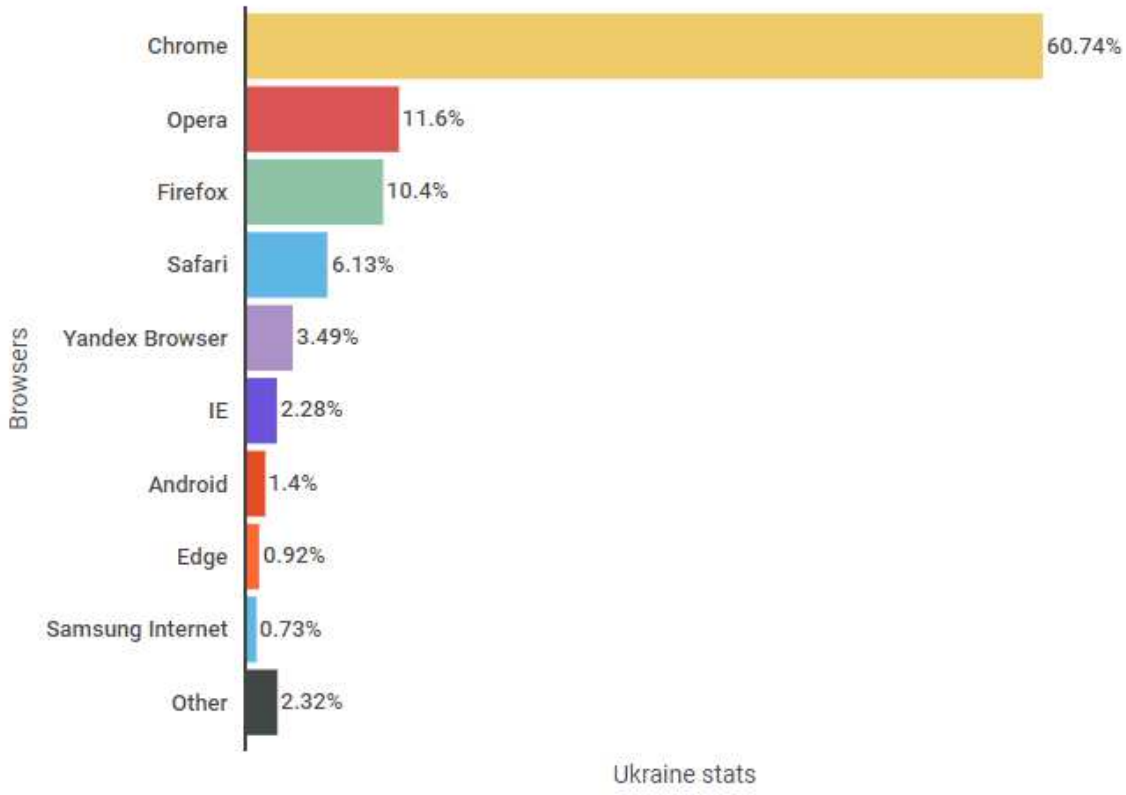

Рисунок 16 – Статистика з використання браузерів в Україні [8][1)]

Крім того, були розглянуті функціональні можливості браузерів, на які користувачі і розробники можуть звернути увагу:

- хмарна синхронізація;
- менеджер завантажень;
- функція приватного перегляду;
- повноекранний режим;
- панель вкладок;
- кастомні плагіни;
- відкритий код.

Порівняння даних особливостей, стосовно розглянутих браузерів, наведені в табл. 1.

[1)] [8] Статистика самых популярных браузеров в мире – Marketer https://marketer.ua/stats-of-browsers-2017/ (дата звернення 22.05.2020)

Таблиця 1 – Порівняння особливостей браузерів

	Chrome	Opera	Firefox	Safari	Yandex	IE	Edge
Наявність хмарної синхронізації	+	+	+	+	+	–	+
Наявність менеджера завантажень	+	+	+	+	+	+	+
Наявність функції приватного перегляду	+	+	+	+	+	+	+
Наявність повноекранного режиму	+	+	+	–	+	+	+
Наявність панелі вкладок	+	+	+	–	+	–	–
Можливість завантаження кастомних плагінів	+	+	+	–	–	+	–
Відкритий код	+	–	+	–	–	–	–

В зв’язку з результатами діаграм та порівнянь було прийнято рішення орієнтуватися на браузер Chrome [9][1)], який був розроблений на основі коду

[1)] [9] Google Chrome – Wikipedia. URL: https://en.wikipedia.org/wiki/Google_Chrome (дата звернення 22.05.2020).

браузера Chromium [10][1)], розробленого компанією Google. Таким чином дозволяючи розробити додаток і для інших браузерів основаних на Chromium. Таких як: Microsoft Edge, Opera, Amazon Silk, Yandex Browser та Vivaldi Browser.

Особливостями браузерів основаних на Chromium є: швидкість, безпека, надійність та розширюваність.

Для досягнення швидкості Chromium було обладнено двигуном відображення веб-сторінок WebKit. Він забезпечує необхідну швидкість рендеринга, маючи при цьому ряд інших переваг. Для обробки JavaScript використовується високопродуктивний двигун V8, що дозволило Chromium стати на момент запуску веб-оглядача одним з найшвидших браузерів в плані обробки JavaScript. Присутнє апаратне прискорення, що забезпечує найвищу продуктивність при обробці динамічного 2D (Canvas) і 3D-контента (WebGL) за допомогою графічного процесора. Для прискорення доступу до сторінок використовується технологія попереднього читання DNS і попередньої відрисовки сторінок (пререндеринг).

Для забезпечення безпеки в Chromium була обрана модель «пісочниці» (спеціально ізольованого середовища), яка давала можливість обмежити простір для атаки призначеного для користувача комп'ютера через використану вразливість. У даній пісочниці працює движок відображення, так-як згідно з дослідженнями Google, 70% загроз працюють саме в движку відображення, який взаємодіє з ненадійним вмістом.

З метою підвищення стабільності для Chromium використовується мультипроцесорна архітектура. Браузер, движок рендеринга, розширення, підключаємі модулі працюють в окремих процесах. Таким чином, при порушенні роботи, наприклад плагіну, браузер продовжить роботу в звичайному режимі, видавши пропозицію про перезапуск цього плагіну.

[1)] [10] Chromium – Wikipedia. URL: https://en.wikipedia.org/wiki/Chromium (дата звернення 22.05.2020).

Але головною особливістю Chromium, для написання плагіна до даного браузеру, є розширюваність. Для забезпечення розширюваності, при написанні розширень можуть використовуватися ті ж технології, що і при написанні веб-сторінок, тобто HTML для створення розмітки, CSS для стилізації і JavaScript для програмування. Також, новітні версії Chromium забезпечують підтримку HTML5 та CSS3, надаючи можливість використовувати такі веб-технології, як Canvas і CSS-анімація.

Використовуючи нативні API, плагіни для Chromium можуть взаємодіяти з закладками, надаючи можливість створювати їх і проводити над ними різні дії; контекстним меню, дозволяючи редагувати його вміст; вкладками, дозволяючи сортувати їх, змінювати і проводити інші дії; іншими нативними API, в тому числі експериментальними. Зі сторонніх API є можливість працювати з DOM, HTML5 API's, WebKit API та іншими.

Також для Chromium можна використовувати крос-браузерий NPAPI-модуль, що дозволяє викликати нативний бінарний код розширення через JavaScript для обміну даними між системою і браузером. Великим недоліком такого використання розширення, написаних за допомогою цього API, є небезпечність. Розширення отримують такі ж права, які має браузер, і через уразливість в розширенні зловмисник може завдати шкоди системі.

Розроблені розширення є можливість викладати в офіційній галереї розширень. Розширення, крім тих що використовують інтерфейс NPAPI, не проходять попередньої перевірки і відразу з'являються в інтернет каталозі. Всі розширення зберігаються в crx-форматі, що є особливо побудованим ZIP-файлом, який можна розпакувати більшістю архіваторами. Згідно оновленої політики безпеки, розробники повинні виплатити внесок в розмірі 5$ для початку публікації розширень в інтернет каталозі. Розширення в Chromium мають можливість оновлюватися самостійно, використовуючи протокол Omaha. Також Chromium має вбудовану підтримку сценаріїв Greasemonkey, що істотно розширює можливості браузера.

Сам плагін браузеру, згідно з [4][1)], Chromium представляє собою набор скриптів. Найголовніший з них – manifest.json. У деяких випадках розширення може складатися тільки з цього файлу – таке може бути у розширення-теми, яке тільки змінює стиль оформлення вікна браузера. У файлі-маніфесті міститься вся інформація про розширення, а також про те, що і коли потрібно запускати, що розширення дозволено робити, до яких ресурсів воно саме дає доступ, і деякі настройки браузера. Решта – скрипти Javascript, файли HTML, файли CSS, а також дані для них (наприклад, файли зображень). Ілюстрацію до даної структури можна побачити на рис. 17.

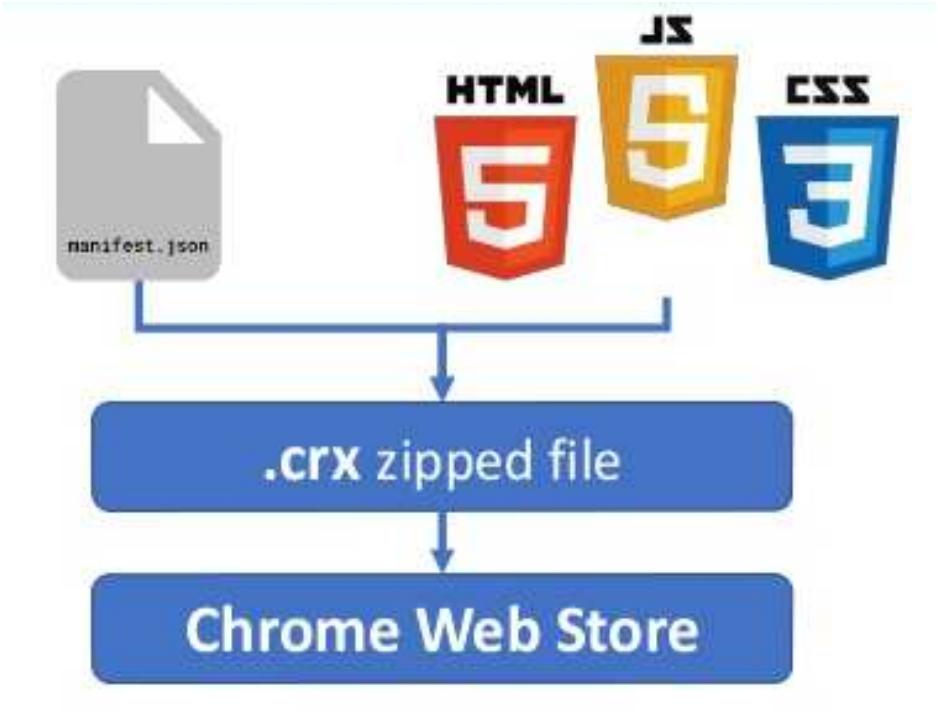

Рисунок 17 – Структура браузерного плагіну типу WebExtension [4][1)]

В зв'язку з інструментами веб-розробки, які будуть використовуватися за типом розробляємого браузерного додатку, буде приведено опис основних засобів веб-розробки.

[1)] [4] Develop Extensions – Google Chrome. URL: https://developer.chrome.com/extensions/devguide (дата звернення 22.05.2020).

HTML [11][1)] – це мова розмітки гіпертексту (від англ. HyperText Markup Language).

Дана мова застосовується для створення веб-сторінок. Код HTML інтерпретується (обробляється) браузером і відображається у вигляді документа в зручній для людини формі.

HTML є невід'ємною складовою і основою практично будь-який вебсторінки. Мова HTML в першу чергу виступає як засіб логічної розмітки сторінки.

Саме HTML дозволяє наділяти вміст сторінки певним змістом, а реалізується це за допомогою тегів.

CSS [12][2)] – це мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки. Назва походить від англ. Cascading Style Sheets – каскадні таблиці стилів.

Додання зовнішнього вигляду документів HTML – це хоч і найпопулярніший, однак лише окремий випадок застосування мови CSS, тому що з його допомогою можна надавати вид і документам інших типів: XHTML, SVG і XUL.

Метою створення CSS було відділення опису логічної структури вебсторінки від її зовнішнього вигляду.

Роздільний опис логічної структури та подання документа дозволяє більш гнучко управляти зовнішнім виглядом документа і мінімізувати обсяг повторюваного коду, який би неминуче виникав при використанні HTML для опису зовнішнього вигляду документа.

За допомогою CSS веб-розробник може задавати для сторінки і окремих її елементів різні гарнітури та розміри шрифту, кольору елементів, відступи елементів один від одного, розташування окремих блоків на сторінці і т.д.

[1)] [11] HTML – Wikipedia. URL: https://en.wikipedia.org/wiki/HTML (дата звернення 22.05.2020)

[2)] [12] Cascading Style Sheets – Wikipedia. URL: https://en.wikipedia.org/wiki/Cascading_Style_Sheets (дата звернення 22.05.2020)

Також для того, щоб використовувати CSS для додання зовнішнього вигляду HTML-документу, потрібно цей документ пов'язати зі стилями, тобто «Повідомити» HTML-документу, що він буде оформлений за допомогою CSS.

Для цього існують різні способи підключення CSS до документа, які дають браузеру знати, що на сторінку в цілому, або до якихось окремих її елементів повинно бути застосовано стильове оформлення.

Таблиці стилів можуть розташовуватися як безпосередньо всередині того, документа, до яких вони будуть застосовуватися, так і перебувати в окремому файлі, що має розширення .css.

Файл з css стилями, або CSS-файл є звичайним текстовим файлом. У ньому пишуться спеціальні інструкції, що описують зовнішній вигляд елемента та його позиціонування на сторінці а також коментарі.

JavaScript [13][1)][14][2)] – це мова програмування, яка знайшла найбільш широке застосування в браузерах для додання інтерактивності веб-сторінок.

Одним з основних завдань JavaScript є маніпулювання елементами DOM-моделі web-сторінки.

DOM є об'єктною моделлю документа (від англ. Document Object Model).

Згідно DOM, документ (наприклад, веб-сторінка) може бути представлений у вигляді дерева об'єктів, що володіють рядом властивостей, які дозволяють виробляти з ним різні маніпуляції:

- отримання вузлів;
- зміна вузлів;
- зміна зв'язків між вузлами;
- видалення вузлів.

[1)] [13] JavaScript – Wikipedia. URL: https://en.wikipedia.org/wiki/JavaScript (дата звернення 22.05.2020).

[2)] [14] Дэвид Флэнаган – JavaScript. Подробное руководство, 6-е 2018 года. Издательство Диалектика; ISBN, 978-1-4919-5202-3.

Саме ці маніпуляції і дозволяють здійснювати над елементами сторінки мова JavaScript.

Існує два способи підключення скриптів JavaScript: посилання на файл в якому храниться код та опис коду в самому файлу, стосовно якого потрібно застосувати цей скрипт. Для обидва способів використовується тег script.

Приклад застосування тегу script:

```
<!DOCTYPE HTML>
<html>
<body>
  <p>Перед скриптом</p>
  <script src='file_name.js'>
    [Код JavaScript]
  </script>
  <p>Після скрипту</p>
</body>
</html>
```

2.2 Вибір середовища розробки

Для розробки даного проекту в якості середовища розробки перевагу було віддано IDE замість спеціалізованого текстового редактора, в зв'язку з тим IDE, чим вона відрізняє від текстового редактора, як правило має всі основні функції для web-розробки, дозволяючи використовувати при розробці тільки одну програму – IDE, таким чином оптимізуючи робочий процес. Вибір між IDE для web-розробки здійснювався між програмами: Eclipse, NetBeans, Geany, Light Table.

Eclipse – вільне інтегроване середовище розробки модульних міжплатформених додатків. Розвивається і підтримується Eclipse Foundation.

Eclipse IDE використовується для розробки ПЗ на багатьох мовах. Можливість підключення різних плагінів, що дозволяє спростити розробку вебдодатків. Підтримує роботу з Java, JavaScript, PHP і іншими мовами, а також створення мобільних додатків. Наявність вбудованого юніт-тестування та оптимізації тестів. Також має подрібни налаштування графічного інтерфейсу.

IDE NetBeans – середовище з відкритим вихідним кодом, світовою спільнотою користувачів і розробників. Це середовище служить для швидкої та легкої розробки настільних, мобільних і веб-додатків на Java, JavaScript, HTML5, PHP, C / C ++ та інших мовах.

Netbeans – середовище, яке надає розробнику аналізатор і редактор коду на Java, а також ряд нових інструментів для HTML5 і JavaScript, в тому числі для Node.js, KnockoutJS і AngularJS.

NetBeans IDE підтримують рефакторинг, профілювання, виділення синтаксичних конструкцій кольором, автоматичне доповнення конструкцій і безліч визначених шаблонів коду.

NetBeans самостійно здійснює відступи, доповнює слова і дужки, робить синтаксичне і семантичне виділення вихідного коду.

Geany – компактне середовище, яке підтримує HTML, XML, PHP і інші мови програмування. Основні можливості:

- підсвічування синтаксису;
- фолдінг (згортання коду);
- автозавершення коду;
- сніппети;
- спливаючі підказки;
- багатомовність;
- таблиця символів;
- навігація по коду;
- готова система для компіляції і виконання коду;
- управління проектами;
- інтерфейс, побудований на плагінах.

Light Table – легковесне інтегроване середовище розробки, написане на ClojureScript з використанням node.js.

Відмінними рисами нової середовища програмування являються простота і мінімалізм інтерфейсу. Таким чином, крім миттєвого показу результату

обчислень, це виражається, наприклад, у відображенні документації по функції при виділенні мишею її назви.

Підтримка додаткових мов програмування можлива за допомогою плагінів.

Більш детальне порівняння IDE наведене у табл. 2.

Таблиця 2 – Порівняння інтегрованих середовищ розробки

	Eclipse	NetBeans	Geany	Light Table
Підсвічування синтаксису	Підтримується			
Налагодження програми	Підтримується		Підтримується у якості окремого модулю	Підтримується
Браузер класів	Підтримується			Відсутній
Рефакторинг	Підтримується		Відсутній	Відсутній
Профілювання	Підтримується		Відсутній	Відсутній
Робота з БД	Підтримується		Відсутній	Відсутній

Таким чином було прийняте рішення обрати інтегроване середовище Eclipse, в зв'язку з його функціональністю а також гнучким інтерфейсом, що дуже важливо при одночасній роботі з декількома мовами під час web-розробці.

2.3 Вибір додаткових засобів та бібліотек

2.3.1 Технологія для запису медіа-пристроїв

Для роботи даного плагіна необхідний доступ і взаємодія з підключеними медіа пристроями, такими як камера, мікрофон, а також спільне використанню екрану. В зв’язку з обраною мовою програмування JavaScript, який

надає доступ до JavaScript API, для вирішення даної проблеми буде використовуватися спеціальний об’єкт navigator.mediaDevices, який реалізує інтерфейс mediaDevices [15][1)], для доступу к медіа-пристроям JavaScript API MediaDevices, за допомогою методів якого можна отримати потоки цих медіа-пристроїв. Слід зазначити що даний інтерфейс повністю налаштовуваний і зручний засіб роботи з медіа-пристроями, тому використання сторонніх допоміжних бібліотек для взаємодії з інтерфейсом не доцільне і не використовувалося. Даний об’єкт для функціонування використовує технологію WebRTC (Web Real-Time Communications) [16][2)], яка дозволяє Web-додаткам, сайтам та плагінам браузеру захоплювати і вибірково передавати аудіо або відео медіа-потоки, а також обмінюватися довільними даними між браузерами, без обов'язкового використання посередників.WebRTC, як зображено на рис. 18, складається з декількох взаємопов'язаних програмних інтерфейсів (API) і протоколів, які працюють разом.

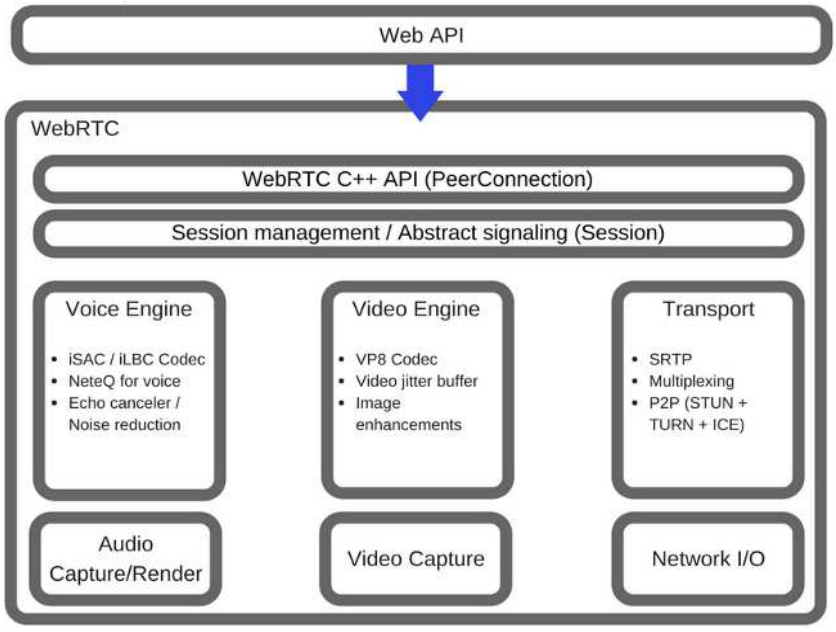

Рисунок 18 – структура технології WebRTC

[1)] [15] MediaDevices - Web APIs | MDN. URL: https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices (дата звернення 22.05.2020)

[2)] [16] Getting started with media devices WebRTC. URL: https://webrtc.org/getting-started/media-devices (дата звернення 22.05.2020)

Найбільш поширений спосіб доступу до медіа-пристроїв, – через функцію getUserMedia(), яка повертає проміс, який отримується для MediaStream до відповідних мультимедійних пристроїв. Ця функція приймає один MediaStreamConstraints об'єкт, який визначає вимоги, які є у розробника.

Виклик getUserMedia() призведе до запиту на дозвіл. Якщо користувач прийме дозвіл, проміс поверне медіа-потік MediaStream, який містить одне відео і одну звукову доріжку. Якщо в дозволі відмовлено, це призведе до помилки PermissionDeniedError. У разі, якщо не підключен відповідний пристрій, спрацює NotFoundError.

У більш складному додатку, для того щоб перевірити всі підключені камери та мікрофони і надати відповідний зворотній зв'язок користувачеві, потрібно викликати функцію enumerateDevices(). Це поверне проміс, який містить масив об’єктів MediaDevicesInfo, що описують кожний відомий мультимедійний пристрій. Дана можливість може використовуватися для того щоб представити інтерфейс, який дозволить користувачеві обрати пристрій, який вони вважають за краще. Кожен MediaDevicesInfo містить властивість з ім'ям kindco з значенням audioinput, audiooutput або videoinput, таким чином вказуючи на тип пристрою.

Більшість комп'ютерів підтримують підключення різних пристроїв під час виконання. Це може бути веб-камера, підключена через USB, гарнітура Bluetooth або набір зовнішніх динаміків. Щоб належним чином підтримувати це, веб-додаток повинен прослуховувати зміни медіа-пристроїв. Дана задача вирішується за допомогою додавання прослуховувача navigator.mediaDevices для devicechange події.

Об'єкт обмежень, який повинен реалізовувати MediaStreamConstraints інтерфейс, який передається в якості параметра getUserMedia(), дозволяє відкривати мультимедійний пристрій, який відповідає певній вимогі. Ця вимога може бути вільно визначено (аудіо або відео) або дуже специфічно (мінімальне розширення камери або точний ідентифікатор пристрою).

Для настройки визначеного пристрою, в додатку, що використовують getUserMedia() API, спочатку визначаються існуючі пристрої, а потім вказуються параметри до цих пристроїв, які відповідають deviceId полю в запиті. Пристрої, якщо можливо, будуть налаштовані відповідно до вказаних параметрів. Також можливо включити ехоподавлення на мікрофонах або встановити визначену або мінімальну ширину і висоту відео з камери.

Як тільки медіа-пристрій було відкрито і буде присутній MediaStream доступ, є можливість призначити відео або аудіо потік елементу для локального відтворення потоку.

HTML-код, необхідний для типового елемента відео, використовуваного з getUserMedia(), зазвичай має атрибути autoplay і playsinline. autoplay атрибут при виклику нових потоки, призначених елементу, будуть автоматичного відтворенні. playsinline атрибут дозволяє відео відтворюватися не знаходячись в повноекранному режимі.

Також для прямих трансляцій є можливість приховати елементи керування за допомогою атрибута controls.

2.3.2 Вибір JavaScript фреймворку

Крім самої реалізації функціоналу плагіну для даного розширення є потрібним засіб для оптимізації користувацького інтерфейсу за допомогою автоматизації алгоритмів та полів додатку.

Для даних завдань, згідно з [17][1)], існують такі популярні фреймворки:

- «React»;
- «Vue»;
- «Angular».

[1)] [17] React или Angular или Vue.js — что выбрать? / Хабр. URL: https://habr.com/ru/post/476312/ (дата звернення 22.05.2020)

React [18][1)] – це бібліотека JavaScript, яка використовується для створення призначеного для користувача інтерфейсу. React був створений компанією Facebook, а перший реліз бібліотеки побачив світ у березні 2013 року.

Спочатку React призначався тільки для веб-розробки, проте пізніше з'явилася платформа React Native, яка вже призначалася для мобільних пристроїв.

React представляється ідеальний інструмент для створення масштабованих веб-додатків (в даному випадку мова йде про фронтенд), особливо в тих ситуаціях, коли додаток являє SPA (односторінковий додаток).

React відносно простий в освоєнні, має зрозумілий та лаконічний синтаксис.

Іншою відмінною рисою бібліотеки є концентрація на компонентах – є можливість створити окремі компоненти і потім їх легко переносити з проекту в проект.

Ще одна особливість React – використання JSX. JSX представляє комбінацію коду JavaScript і XML і надає простий і інтуїтивно зрозумілий спосіб для визначення коду візуального інтерфейсу.

Переваги React:

- легко вивчити, завдяки простому дизайну, використання JSX (HTML-подібний синтаксис) для шаблонів і дуже докладної документації; розробники можуть витрачати більше часу на написання сучасного JavaScript і менше турбуватися про код, специфічного для фреймворка;
- дуже швидкий, завдяки реалізації React Virtual DOM і різним оптимізаціям рендеринга;
- відмінна підтримка рендеринга на стороні сервера, що робить його потужною платформою для контент-орієнтованих додатків;
- першокласна підтримка Progressive Web App (PWA) завдяки генератору додатків `create-react-app`;

[1)] [18] React – A JavaScript library for building user interfaces. URL: https://en.reactjs.org/ (дата звернення 22.05.2020).

- прив'язка даних є односторонньою, що означає менше небажаних побічних ефектів;
- «Redux», найпопулярніша платформа для управління станом додатків в React, її легко вчити і використовувати;
- «React» реалізує концепції функціонального програмування (FP), створюючи простий в тестуванні і багаторазово використовуваний код;
- додатки можуть бути створені за допомогою TypeScript або Facebook's Flow, що мають вбудовану підтримку JSX;
- перехід між версіями, як правило, дуже простий.

Недоліки React:

- «React» не однозначний і залишає розробникам можливість вибирати кращий спосіб розвитку проекту;
- «React» відходить від компонентів на основі класів, що може стати перешкодою для розробників, яким більш комфортно працювати з об'єктно-орієнтованим програмуванням (ООП);
- змішування шаблонів з логікою (JSX) може бути не зручним для деяких розробників при перших знайомствах з React.

Компанії, які використовують React: Facebook, Instagram, Netflix, New York Times, Yahoo, Khan Academy, Whatsapp, Codecademy, Dropbox, Airbnb, Asana, Atlassian, Intercom, Microsoft, Slack, Storybook і багато інших.

Vue.js [19][1)] представляє сучасний прогресивний фреймворк, написаний на мові JavaScript і призначений для створення веб-додатків клієнтського рівня. Основна сфера застосування даного фреймворку – це створення і організація користувацького інтерфейсу.

Vue.js має досить невеликий розмір – не більше 20 кБ, та при цьому володіє хорошою продуктивністю в порівнянні з такими фреймворками як Angular або React.

[1)] [19] Vue.js URL: https://vuejs.org/ (дата звернення 22.05.2020).

Ще однією особливістю Vue є наявність унікального шаблону управління станом, що також є бібліотекою для додатків Vue.js – Vuex. Він служить централізованим сховищем для всіх компонентів програми з правилами, що гарантують, що стан може бути змінено тільки передбачуваним чином. Більш докладне описання шаблону Vuex зображено на рис. 19.

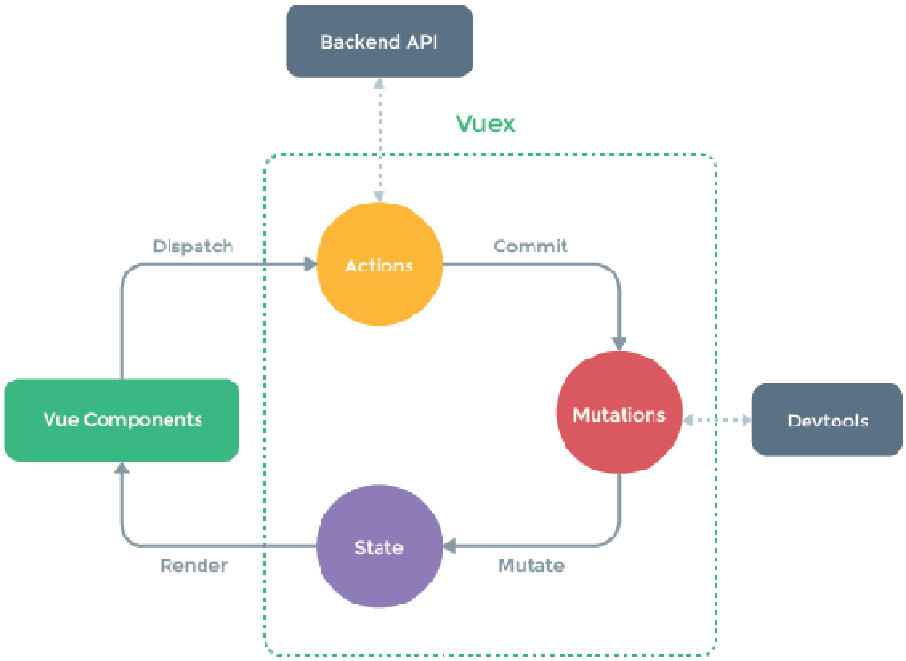

Рисунок 19 – структура шаблону Vuex [20][1)]

Одним з ключових моментів в роботі Vue.js є віртуальний DOM. Віртуальний DOM представляє легковажну копію звичайного DOM. Якщо додатку потрібно дізнатися інформацію про стан елементів, то відбувається звернення до віртуального DOM. Якщо дані, які використовуються в додатку Vue.js, змінюються, то зміни спочатку вносяться в віртуальний DOM. Потім Vue вибирає мінімальний набір компонентів, для яких треба виконати зміни на вебсторінці, щоб реальний DOM відповідав віртуальному.

[1)] [20] What is Vuex? | Vuex URL: https://vuex.vuejs.org/ (дата звернення 22.05.2020)

Vue.js також підтримується всіма браузерами, які сумісні з ECMAScript5. На даний момент це все сучасні браузери, в тому числі IE11.

Переваги Vue.js:

- посилений HTML; це означає, що Vue.js має багато характеристик схожих з Angular, а це, завдяки використанню різних компонентів, допомагає оптимізації HTML- блоків;
- детальна документація; Vue.js має дуже детальну документацію, яка може прискорити процес навчання для розробників і заощадити багато часу на розробку програми, використовуючи тільки базові знання HTML і JavaScript;
- адаптивність; може бути здійснений швидкий перехід від інших фреймворків до Vue.js через схожість з Angular і React з точки зору дизайну і архітектури;
- приголомшлива інтеграція; Vue.js можна використовувати як для створення односторінкових додатків, так і для більш складних веб-інтерфейсів додатків; важливо, що невеликі інтерактивні елементи можна легко інтегрувати в існуючу інфраструктуру без негативних наслідків;
- масштабування; Vue.js може допомогти в розробці досить великих шаблонів багаторазового використання, які можуть бути зроблені майже за той же час, що і більш прості;
- крихітний розмір; Vue.js важить близько 20 КБ, зберігаючи при цьому свою швидкість і гнучкість, що дозволяє досягти набагато кращої продуктивності в порівнянні з іншими платформами.

Недоліком Vue.js є ризик надмірної гнучкості. Іноді у Vue.js можуть виникнути проблеми при інтеграції в величезні проекти, і поки ще немає досвіду можливих рішень.

Компанії, які використовують Vue.js: Xiaomi, Alibaba, WizzAir, EuroNews, Grammarly, Gitlab і Laracasts, Adobe, Behance, Codeship, Reuters.

Angular [21][1)] представляє фреймворк від компанії Google для створення клієнтських додатків. Перш за все він націлений на розробку SPA-рішень (Single Page Application), тобто односторінкових додатків. В цьому плані Angular є спадкоємцем іншого фреймворка AngularJS. У той же час Angular це не нова версія AngularJS, а принципово новий фреймворк.

Angular надає таку функціональність, як двостороннє зв'язування, що дозволяє динамічно змінювати дані в одному місці інтерфейсу при зміні даних моделі в іншому, шаблони, маршрутизація і т.д.

Однією з ключових особливостей Angular є те, що він використовує в якості мови програмування TypeScript.

Переваги Angular:

- «Angular» використовується разом з Typescript; він має виняткову підтримку для цього;
- «Angular-language-service»- забезпечує інтелектуальні можливості і автозаповнення шаблону HTML-компонента;
- нові функції, такі як generation Angular, що використовують бібліотеки npm з CLI, generation, і розробка компонентів, що використовує Angular;
- детальна документація, що дозволяє розробнику отримати всю необхідну інформацію; однак документація вимагає багато часу для навчання;
- одностороння прив'язка даних, яка забезпечує виняткову поведінку додатка, що зводить до мінімуму ризик можливих помилок;
- «MVVM» (Model-View-ViewModel), яка дозволяє розробникам працювати окремо над одним і тим же розділом програми, використовуючи один і той же набір даних;
- впровадження залежностей від компонентів, пов'язаних з модулями і модульність в цілому;

[1)] [21] AngularJS — Superheroic JavaScript MVW Framework. URL: https://angularjs.org/ (дата звернення 22.05.2020).

- структура і архітектура, спеціально створені для великої масштабованості проекту.

Недоліки Angular:

- різноманітність різних структур (Injectables, Components, Pipes, Modules і т. Д.) ускладнює вивчення в порівнянні з React і Vue.js, у яких є тільки «Component».
- відносно повільна продуктивність, враховуючи різні показники; з іншого боку, це можна легко вирішити, використовуючи так званий «ChangeDetectionStrategy», який допомагає вручну контролювати процес рендеринга компонентів.

Компанії, які використовують Angular: Microsoft, Autodesk, MacDonald's, UPS, Cisco Solution Partner Program, AT & T, Apple, Adobe, GoPro, ProtonMail, Clarity Design System, Upwork, Freelancer, Udemy, YouTube, Paypal, Nike, Google, Telegram, Weather, iStockphoto, AWS, Crunchbase.

Таким чином були проаналізовані всі переваги та недоліки фреймворків, звертаючи в першу чергу на таки параметри як швидкість, гнучкість, компактність та документацію, було обрано фреймворк Vue.js

3 РОЗРОБКА ПРОГРАМИ

В даному розділі буде описана розробка плагіну з запису медіа-пристроїв. Етапами для цього є розробка функціоналу додатку та розробка користувальницького інтерфейсу, для взаємодії з функціоналом.

Докладніше про ресурси, які використовуються при розробці додатку зображено на рис. 20.

Докладніше про структуру розділу «Розробка програми» та зв’язок його етапів зображено на рис. 21.

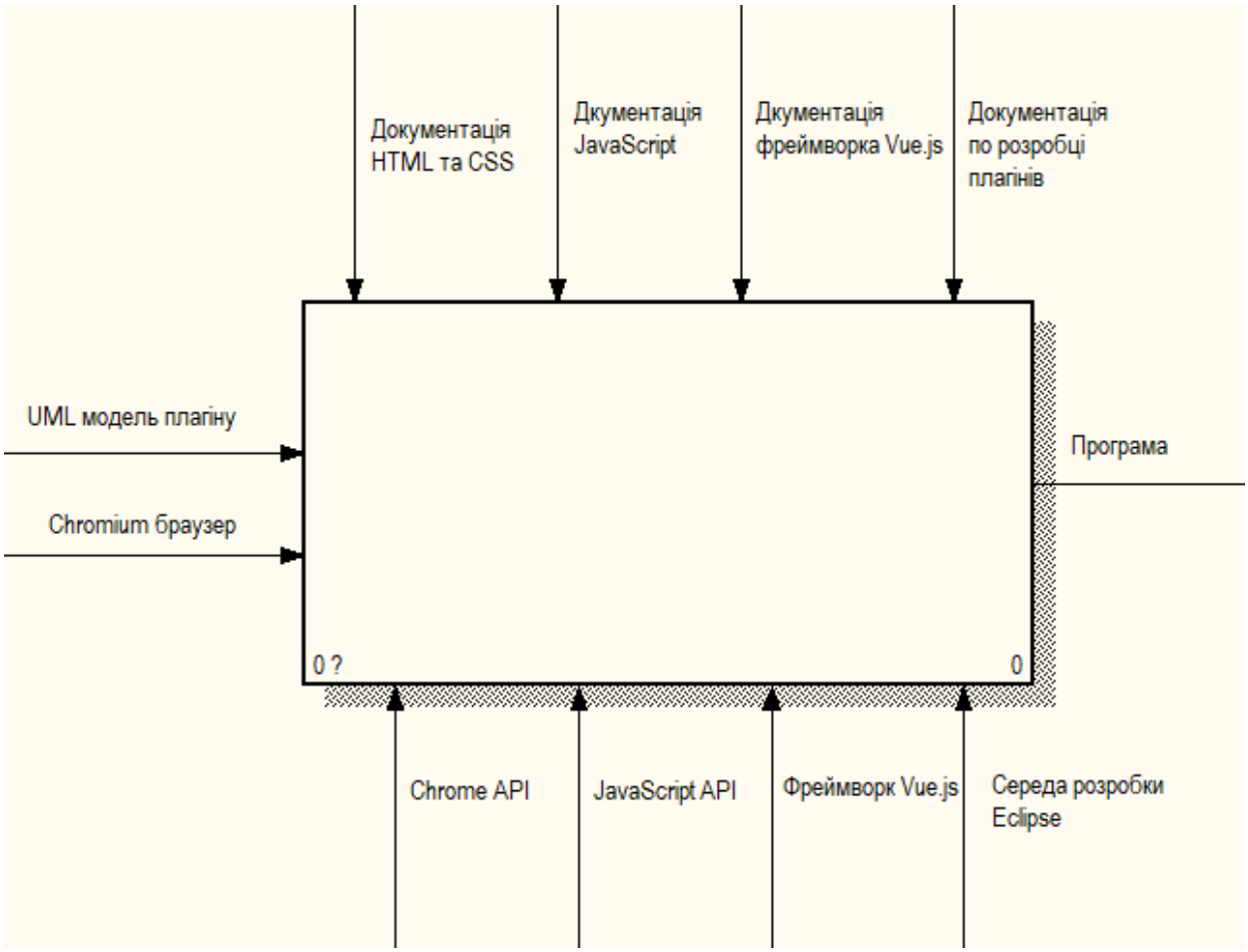

Рисунок 20 – Використані ресурси при розробці додатку

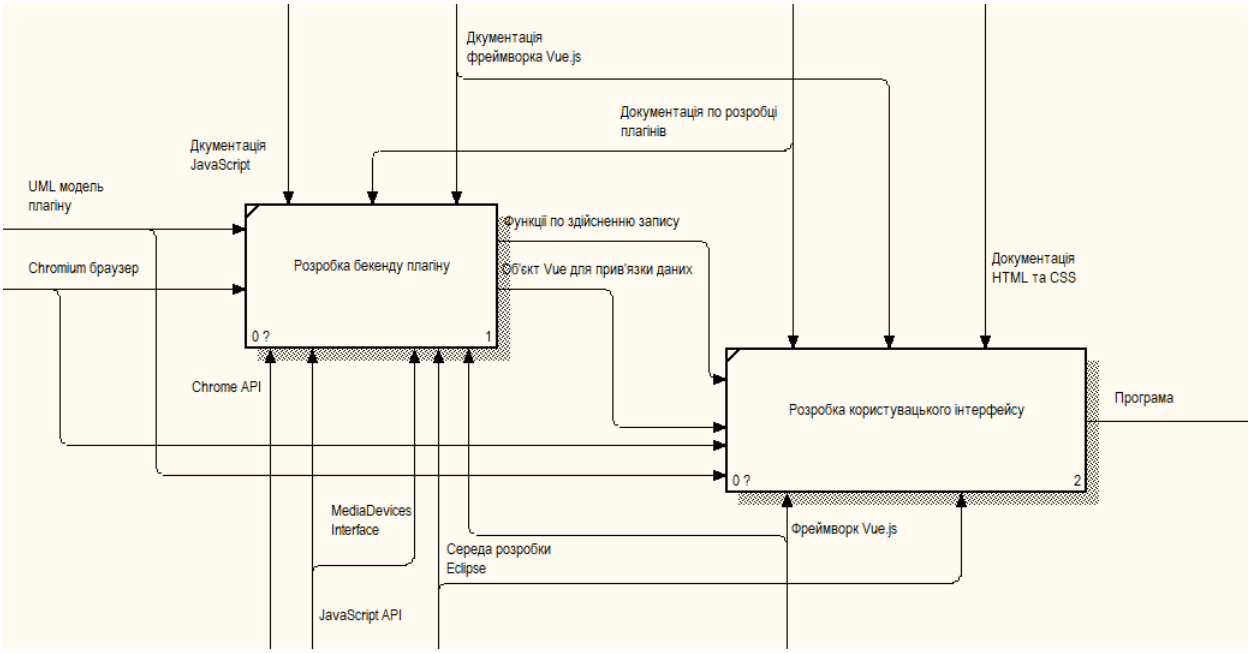

Рисунок 21 – Структурні елементи розробки додатку

3.1 Розробка бекенду плагіну

При розробці проекту був створений об'єкт для зберігання основних даних і налаштувань програми:

```
appData = {
  disableRecord: false,
  recordStatus: 'ready',
  selectedRecordOption: null,
  selectedSoundOption: null,
  requestSharePending: false,
  countDownPending: false,
  countDownNumber: null,
  visibleSoundSelector: false,
  recordTime: 0,
  interval: null,
  audioColumnHeight: 0,
  saveTypes: [
     { label: 'webm', value: 'video/webm;codecs=vp8' },
     { label: 'mp4', value: 'video/mp4;codecs=avc1.4d002a'
     }
  ],
  saveType: null,
  cameraResolutions: [
     { label: '1920p', width: '1920', height: '1080' },
     { label: '1280p', width: '1280', height: '720' },
     { label: '480p', width: '480', height: '270' },
     { label: '360p', width: '360', height: '202' },
```

```
        { label: '240p', width: '240', height: '135' }
      ],
      cameraResolution: null,
      videoframeRates: ['30','25','15'],
      videoframeRate: null,
      openSetting: false,
      audioInputDevices: [],
      audioOutputDevices: [],
      videoInputDevices: [],
      selectedAudioInput: null,
      selectedVideoInput: null,
      openSettingItem: '',
    };
```

disableRecord – поле логічного типу, вказує на факт здійснення в даний момент записи.

recordStatus – поле текстового типу, вказує на стадію записи.

selectedRecordOption – поле об'єкт, що містить інформацію про параметр запису.

selectedSoundOption – поле об'єкт, що містить інформацію про параметр звуку.

countDownPending – логічне поле, що вказують на вчинення відліку.

countDownNumber – поле числового типу, що вказують на початок відліку.

recordTime – час в мілі секундах від початку запису.

interval – поле числового типу, що вказують на інтервал візуального елементу.

saveType – поле текстового типу, що вказують на тип збереженого файлу.

cameraResolution – поле об'єкт, що вказують на дозвіл в якому записується зображення з камери.

videoframeRate – поле об'єкт, що вказують на частоту кадрів.

openSettingItem – поле об'єкт, що вказують на налаштування запису.

Для здійснення користувачем, шляхом Chrome API, вибору цільового вікна для запису був створений об'єкт (докладніше див. додаток А):

```
screenShare = {
  shareId: null,
  requestShare();
  cancelShare() ;
};
```

shareID – ідентифікатор об’єкту з здійснення вибору.

requestShare() – метод, який пропонує користувачеві обрати ціль запису.

cancelShare() – метод, який закриває вікно здійснення вибору.

Для здійснення запису медіа пристроїв та екрану був створений об'єкт з основним функціоналом для цього (докладніше див. додаток А):

```
recorder = {
  scrShare;
  stream;
  start();
  recordTab();
  recordScreen();
  recordCamera();
}
```

scrShare – об’єкт з фунціоналом вибору цільового вікна запису.

stream – відео-потік.

start() – функція початку запису, яка згідно з обраними опціями налаштовує та вибирає тип запису.

recordTab() – функція початку запису вкладки браузера.

recordScreeen() – функція початку запису екрану комп’ютера.

recordCamera() – функція початку запису камери.

Для взаємодії скриптів контенту а також скриптом інтерфейсу з об'єктом для здійснення запису, був створений відповідний для цього об'єкт (докладніше див. додаток А):

```
mediaRecorder = {
```

```
  recorder;
  init();
  bindEventStream();
  start();
  pause();
  resume();
  stop();
  downloadVideo();
  closeStream();
}
```

Recorder – об’єкт для здійснення процесу запису.

Init() – функція ініціалізації всіх параметрів для здійснення запису.

bindEventStream() – функція встановлення подій, для прослуховування повідомлень від скрипту інтерфейсу користувача.

start() – початок запису.

pause() – призупинення запису.

resume() – відновлення запису.

stop() – припинення запису.

downloadVideo() – збереження записаного файлу.

closeStream() – закриття відео-потоку.

3.2 Розробка користувальницького інтерфейсу

Так як для створення інтерфейсу використовувався Vue.js, то для автоматизованості інтерфейсу був створен об'єкт додатка – об'єкт Vue. Цей об'єкт, по-перше, визначає кореневий елемент додатки на веб-сторінці за допомогою параметра el. Також об'єкт визначає використовувані дані через параметр data. Останнім параметром об'єкта Vue є methods – визначає дії, які виконуються в додатку. Таким чином дозволяючи для плагіну автоматично застосовувати зміни значень фонового скрипту до інтерфейсу.

Неповний код Vue об'єкта (докладніше див. додаток Б):

```
app = new Vue({
  el: '#app',
  data: appData,
```

```
computed: {
  recordTimeFormat();
},
created();
methods: {
  init();
  startRecord();
  pauseRecord();
  stopRecord();
  resumeRecord();
  selectOption(data);
  selectSound(option);
  toggleSound();
  toggleCamera();
  toggleSoundSelector();
  selectAudioInput(item);
  selectVideoInput(item);
  setDisableRecord();
  cancelShare();
  setConfig();
  onListClick();
  getAllDevices();
  notification(options);
}});
```

recordTimeFormat() – функція, яка здійснює переклад милі секунд в зрозумілий для користувача час, яке відображається на панелі інтерфейсу.

created() – подія, яка виникає в разі створення об’єкту.

init (), startRecord(), pauseRecord(), stopRecord(), resumeRecord() – функції контролю процесу запису. Для взаємодії з скриптами бекенда використовуються повідомлення.

selectOption(), selectSound(), selectAudioInput(), selectAudioInput() – функції установки опцій для процесу запису.

setConfig() – функція збереження налаштувань додатку зроблених користувачем.

onListClick() – функція, що відповідає за обробку події відкриття списку.

getAllDevices() – функція повертає список всіх записуючих пристроїв підключених до комп’ютера.

notification() – функція, яка повідомляє користувача.

cancelShare() – функція, яка закриває вікно вибору цільового вікна для запису.

toggleSound(), toggleCamera(), toggleSoundSelector() – функції обробки перемикача опцій додатку.

В елементах на веб-сторінці, завдяки Vue об'єкту, є можливість виведення значень властивостей і більш того зв'язування ділянок веб-станиці з цим елементом, для цього використовуючи подвійні фігурні дужки.

Частина коду HTML спливаючого вікна додатку, який використовує об’єкт Vue (докладніше див. додаток В):

```
<div class="setting-item">
     <label   class="setting-item-label"   data-i18n="Type">
     </label>
     <div  class="select-options"  style="width:7em"  v-if=
     "saveTypes.length"  :class="{open:  openSettingItem  ===
     'type'}">
               <p class="option-current" @click.stop= "open
          SettingItem  ===  'type'  ?  openSettingItem  =  ''  :
          openSettingItem ='type'" :title="saveType.label">
               {{saveType.label}}
          </p>
          <ul class="options-list">
                    <li class="options-item" v-for="item in
               saveTypes"      @click="selectSaveType(item)"
               :class=  "{  active:  item  ===  saveType  }">
               {{item.label}}</li>
          </ul>
     </div>
</div>
```

Наведений фрагмент коду являє собою настройку, у вигляді випадаючого списку, для вибору типу відеофайлу при його збереженні.

4 ТЕСТУВАННЯ

Під час тестування були перевірені на працездатність навігація та взаємодія з інтерфейсом додатку (докладніше див. рис. 22).

Були перевірені на працездатність опції (докладніше див. рис. 23).

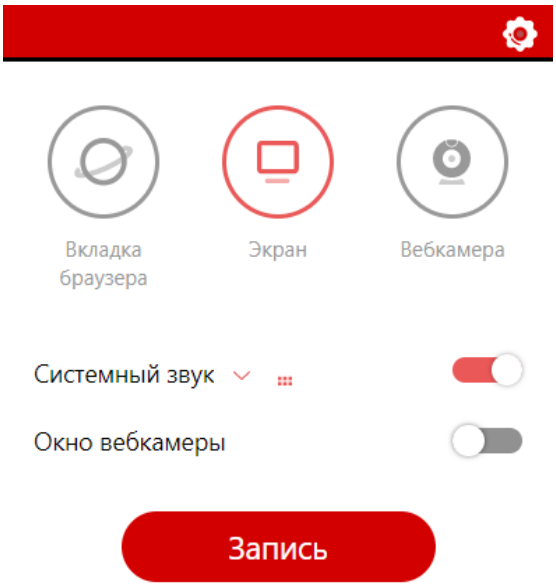

Рисунок 22 – Інтерфейсу плагіну

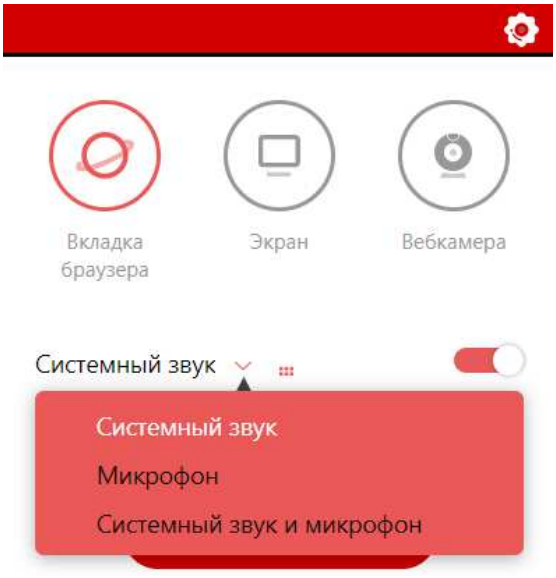

Рисунок 23 – Додаткові опції інтерфейсу плагіну

Були перевірені на працездатність настройки додатку. Зокрема їх локальному збереженню (докладніше див. 24).

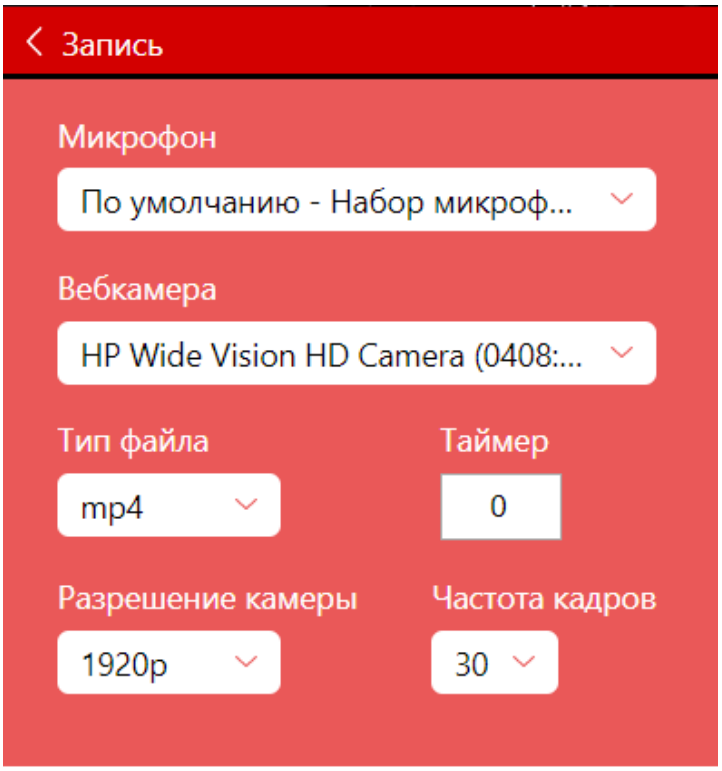

Рисунок 24 – Вкладка з настройками плагіну

Також було перевірне сам процес запису та збереженню записного відео-файлу (Докладніше див. рис 25, 26).

Рисунок 25 – Інтерфейс регулювання процесу запису

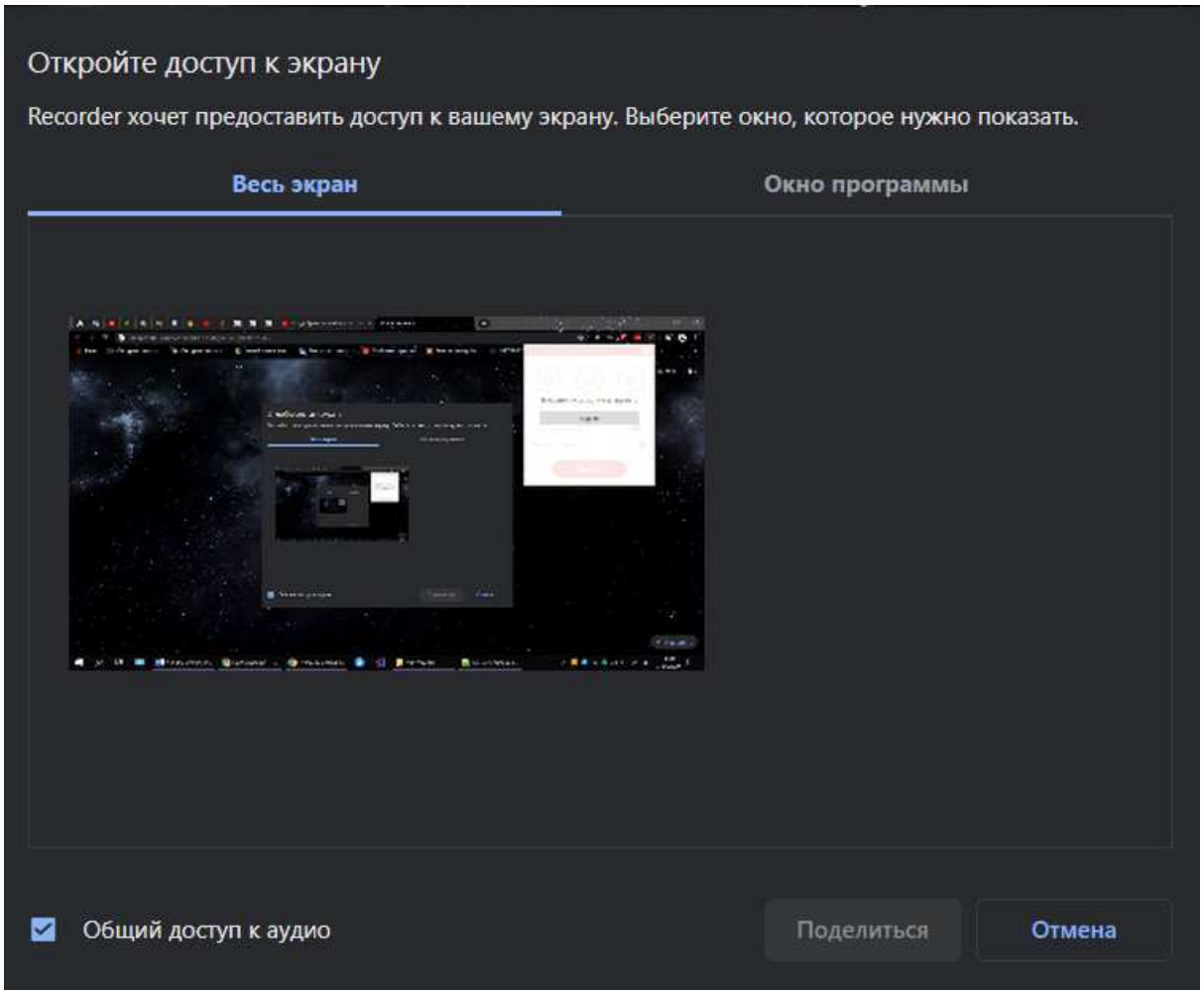

Рисунок 26 – Вікно вибору цільового вікна для запису

У результаті тестування додатку помилок виявлено не було. Всі функції та елементи плагіну працездатні. Додаток задовольняє всі поставлені перед ним вимоги.

ВИСНОВКИ

При розробці даного проекту були виконані наступні кроки:

- аналізу предметної області;
- вибір засобів застосування;
- розробки програми;
- тестування програми.

В результаті успішного виконання розробки була отримана програма, яка містить в собі функції запису екрану, запису приймальних пристроїв звуку, запису відеокамери та має зручний інтерфейс, іншими словами задовольняє всі поставлені перед нею вимоги.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. How screen recording can be used in different ways [Sponsored] – Memeburn. URL: https://memeburn.com/2018/04/screen-recording-can-used-different-ways/ (дата звернення 22.05.2020).
2. Developments in audio-visual education. URL: https://unesdoc.unesco.org/ark:/48223/pf0000001433 (дата звернення 22.05.2020).
3. What's an Internet Browser? A Layman's Guide. URL: https://www.thebalanceeveryday.com/what-is-internet-browser-892819 (дата звернення 22.05.2020).
4. Develop Extensions – Google Chrome. URL: https://developer.chrome.com/extensions/devguide (дата звернення 22.05.2020).
5. Что такое расширения браузера? Их виды и польза. URL: https://evergreens.com.ua/ru/articles/browser-extensions.html (дата звернення 22.05.2020).
6. Nimbus Note - Самое удобное приложение для создания, редактирования и хранения заметок. URL: https://nimbusweb.me/ (дата звернення 22.05.2020).
7. Screencastify | The #1 Screen Recorder for Chrome/URL: https://www.screencastify.com/ (дата звернення 22.05.2020).
8. Статистика самых популярных браузеров в мире – Marketer https://marketer.ua/stats-of-browsers-2017/ (дата звернення 22.05.2020)
9. Google Chrome – Wikipedia. URL: https://en.wikipedia.org/wiki/Google_Chrome (дата звернення 22.05.2020).
10. Chromium – Wikipedia. URL: https://en.wikipedia.org/wiki/Chromium (дата звернення 22.05.2020).

11. HTML – Wikipedia. URL: https://en.wikipedia.org/wiki/HTML (дата звернення 22.05.2020).
12. Cascading Style Sheets – Wikipedia. URL: https://en.wikipedia.org/wiki/Cascading_Style_Sheets (дата звернення 22.05.2020).
13. JavaScript – Wikipedia. URL: https://en.wikipedia.org/wiki/JavaScript (дата звернення 22.05.2020).
14. Дэвид Флэнаган – JavaScript. Подробное руководство, 6-е 2018 года. Издательство Диалектика; ISBN, 978-1-4919-5202-3.
15. MediaDevices - Web APIs | MDN. URL: https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices (дата звернення 22.05.2020).
16. Getting started with media devices WebRTC. URL: https://webrtc.org/getting-started/media-devices (дата звернення 22.05.2020).
17. React или Angular или Vue.js — что выбрать? / Хабр. URL: https://habr.com/ru/post/476312/ (дата звернення 22.05.2020).
18. React – A JavaScript library for building user interfaces. URL: https://en.reactjs.org/ (дата звернення 22.05.2020).
19. Vue.js URL: https://vuejs.org/ (дата звернення 22.05.2020).
20. What is Vuex? | Vuex URL: https://vuex.vuejs.org/ (дата звернення 22.05.2020).
21. AngularJS — Superheroic JavaScript MVW Framework. URL: https://angularjs.org/ (дата звернення 22.05.2020).

ДОДАТОК А

Основний код background скриптів

```
window.screenShare = {
  shareId: null,
  requestShare() {
    const options = ['screen', 'window'];
           if (appData.audioOutputDevices.length &&
           isRecordSystemSound()) {
      sources.push('audio');
    }

    this.shareId = chrome.desktopCapture.chooseDesktopMedia
    (options, settedTab(), streamId => {
      if (streamId) {
        createStream(streamId);
      } else {
        sendMessage({action:'reject-share'});
      }
      appData.requestSharePending = false;
    });
  },
  cancelShare(){
     chrome.desktopCapture.cancelChooseDesktopMedia(
     this.shareId);
     appData.requestSharePending = false;
  }
};


window.mediaRecorder = {
  recorder = null,
  streams: [],
  init(stream, stream_scr) {
    stream = stream;
    streams.push(stream, stream_scr);
    this.bindEventStream(streams);
    instance = new MediaRecorder(stream);

    instance.onAvaibleData = function(e) {
      var recorderHints = appData.selectedRecordOption.
      recorderHints;
      var options = {
        type: window.appData.saveType.value
      };
      mediaRecorder.downloadVideo(new Blob([e.data],
      options));
    };

      instance.onStop = function(e) {
      mediaRecorder.closeStream(mediaRecorder.streams);
    };

      instance.onError = function(error) {
      console.log('error', error)
    };
```

```
},
     bindEventStream(streams) {
  if (!Array.isArray(streams)) {
    streams = [streams];
  }
  streams.forEach(stream => {
    if (stream) {
      addStreamStopListener(stream, function() {
        mediaRecorder.stop();
        sendMessage({action:'app-init'});
        appData.recordStatus = 'ready';
        cameraPreview.closeCameraPreview();
      });
    }
  });
},
startRecording() {
  instance.start();
  stopwatch.start();
  sendMessage({action:'start-record'});
  appData.recordStatus = 'recording';

  console.log('recording');
},
pauseRecording () {
  instance.pause();
  stopwatch.pause();
},
resumeRecording () {
  instance.resume();
  stopwatch.resume();
  popup.sendMessage('resume-record');
},
stopRecording () {
  if (instance && instance.state !== 'inactive') {
    instance.stop();
  }

  stopwatch.stop();

  tabAudio.stop();
},
VDownload (videoBlob) {
   let blobUrl = URL.createObjectURL(videoBlob);
   var downloadBtn = document.createElement('a');
   downloadBtn.download = '';
   downloadBtn.href = blobUrl;
   document.body.appendChild(downloadBtn);
   downloadBtn.click();

   chrome.downloads.download({
     url: blobUrl,
     saveAs: true
   });
},
closeStream(streams) {
  if (!Array.isArray(streams)) {
    streams = [streams];
```

```
    }
    streams = [...streams];
    streams.forEach(stream => {
      if (stream) {
        if (typeof stream.stop === 'function') {
          stream.stop();
        } else {
          stream.getTracks().forEach(track => {
            if (typeof track.stop === 'function') {
              track.stop();
            }
          });
        }
      }
    });
  }
};

window.recorder = {
       scrShare: null,
  stream: null,

  start() {
    mediaRecorder.streams = [];
    if (appData.selectedRecordOption.value === 'browserTab'
     ) {
      this.recordTab();
      } else if (appData.selectedRecordOption.value ===
      'screen') {
      this.recordScreen();
       } else if (appData.selectedRecordOption.value ===
       'camera') {
       this.recordCamera();
       }
  },
  recordTab() {
    chrome.tabCapture.capture(
      {
        audio: isRecordSystemSound(),
        video: true,
        videoConstraints: {
          mandatory: {
            chromeMediaSource: 'tab',
            maxWidth: 1920,
            maxHeight: 1080,
          }
        },
      },
      function(mediaStream) {

        if (isRecordSystemSound()) {
          tabAudio.init(mediaStream);
          tabAudio.play();
        }
        if (isRecordMicrophone()) {
          navigator.mediaDevices
            .getUserMedia({ audio: true })
            .then(audioStream => {
              mediaRecorder.streams.push(mediaStream);
```

```
                var streamclone= mediaStream.clone();
                     var streamplusaudio = getStreamplusaudio
                     ([streamclone, audioStream]);
                mediaRecorder.streams.push(streamplusaudio);
                if (streamplusaudio && streamplusaudio.
                  getAudioTracks().length) {
                  var mixedTrack = streamplusaudio.
                       getAudioTracks()[0];
                  streamclone.addTrack(mixedTrack);

                  streamclone.getAudioTracks().forEach(
                  function(track) {
                    if (track === mixedTrack) return;
                    streamclone.removeTrack(track);
                  });
                }
                     startRecordingScreen(streamclone,
                     audioStream
                 );
              })
              .catch(error => {
                console.log(error);
              });
          } else {
            startRecordingScreen(mediaStream);
          }
        }
      );
    },
    recordScreen() {
      screenShare.requestShare();
    },
    recordCamera() {
      var constraints = {
        video: true,
        audio: isRecordMicrophone()
      };

      navigator.mediaDevices
        .getUserMedia(constraints)
        .then(mediaStream => {

          startRecordingScreen(mediaStream);
        })
        .catch(error => {
          console.log(error);
        });
    }
  };
```

ДОДАТОК Б

Код об’єкту Vue

```
const app = new Vue({
  el: '#app',
  data: appData,
  computed: {
    recordTimeFormat() {
      let hour = Math.floor(this.recordTime / 3600);
      let min = Math.floor((this.recordTime % 3600) / 60);
      let sec = (this.recordTime % 3600) % 60;

      hour = hour > 9 ? hour : '0' + hour;
      min = min > 9 ? min : '0' + min;
      sec = sec > 9 ? sec : '0' + sec;

      return hour + ':' + min + ':' + sec;
    }
  },
  created() {
    this.visibleSoundSelector = false;
    if (!this.selectedRecordOption) {
      this.selectedRecordOption = this.recordOptions[0];
    }
    if (!this.selectedSoundOption) {
     this.selectedSoundOption =
     this.selectedRecordOption.soundOptions[0];
    }

if(!this.saveType){
     this.saveType = this.saveTypes[0];
     chrome.storage.local.get(['saveType'], function(store)
     {
          if (store.saveType) {
               app.$data.saveType = app.$data.
               saveTypes[store.saveType];
          }
     });
}
if(!this.constcountDownNumber){
     this.constcountDownNumber = 0;


chrome.storage.local.get(['constcountDownNumber'],
          function(store) {
          if (store.constcountDownNumber) {
               app.$data.constcountDownNumber =   store.
               constcountDownNumber;
          }
     });
}
if(!this.cameraResolution){
     this.cameraResolution = this.cameraResolutions[0];

     chrome.storage.local.get(['cameraResolution'],
     function(store) {
```

```
            if (store.cameraResolution) {
                app.$data.cameraResolution = app.$data.
                cameraResolutions[store.cameraResolution];
            }
        });
    }
    if(!this.videoframeRate){
        this.videoframeRate = this.videoframeRates[0];

        chrome.storage.local.get(['videoframeRate'],
        function(store) {
            if (store.videoframeRate) {
                app.$data.videoframeRate = app.$data.
                videoframeRates[store.videoframeRate];
            }
        });
    }

    this.getAllDevices();

    this.camera = !cameraPreview.isClosed;
    this.openSetting = false;

    this.setDisableRecord();
  },
  methods: {
    init() {
      this.recordStatus = 'ready';
      this.requestSharePending = false;
      this.recordTime = 0;

      this.setDisableRecord();
    },
    start() {
      doAction('start');
      if (this.selectedRecordOption.value === 'screen') {
        this.requestSharePending = true;
      }
    },
    pause() {
      if (this.recordStatus === 'pause') {
        this.recordStatus = 'recording';
        doAction('resume');
      } else {
        this.recordStatus = 'pause';
        doAction('pause');
      }
    },
    stop() {
      doAction('stop');
      this.init();

      if (this.camera) {
        this.toggleCamera();
      }
    },
    startRecord() {
      this.recordStatus = 'recording';
      this.setConfig();
```

```
    },
    resumeRecord() {
      this.startRecord();
    },
    selectOption(data) {
      this.disableRecord = false;
      this.selectedRecordOption = data;
      if (data.value !== 'camera') {
        data.soundOptions.forEach(option => {
          if (option.label === this.selectedSoundOption.
               label) {
               this.selectedSoundOption = option;
          }
        });
      } else {
       data.soundOptions.forEach(option => {
          if (option.label === this.selectedSoundOption.
          label) {
            this.selectedSoundOption = option;
          }
        });
        this.selectedSoundOption = data.soundOptions[1];
      }

      if (this.selectedRecordOption.value === 'camera') {
        doAction('open-camera-view');
      } else if (!this.camera) {
        doAction('close-camera-view');
      }
    },
    selectSound(option) {
      if (option.disable) return;
      this.selectedSoundOption = option;
      this.visibleSoundSelector = false;
    },
    toggleSound() {
      if (this.audioInputDevices.length === 0) {
        this.notification({message:
        chrome.i18n.getMessage('lackDeviceTips')});
        return;
      }
      this.sound = !this.sound;
    },
    toggleCamera() {
      this.camera = !this.camera;
      if (this.camera) {
        doAction('open-camera-view');
      } else {
        doAction('close-camera-view');
      }
    },
    toggleSoundSelector() {
          if
(this.selectedRecordOption.soundOptions.length > 1
           ) {
         this.visibleSoundSelector = !this.
              visibleSoundSelector;
      }
    },
```

```
    selectAudioInput(item) {
      this.selectedAudioInput = item;
      this.openSettingItem = '';

      chrome.storage.local.set({
         selectedAudioInput: this.$data.audioInputDevices.
         findIndex(el => { return el.label ===
         app.$data.selectedAudioInput.label })
      });
    },
    selectVideoInput(item) {
      this.selectedVideoInput = item;
      this.openSettingItem = '';

     chrome.storage.local.set({
       selectedVideoInput:
       this.$data.videoInputDevices.findIndex(el => {
       return el.label===app.$data.selectedVideoInput.label
     })
     });
    },

selectSaveType(item) {
      this.$data.saveType = item;
      this.openSettingItem = '';

  chrome.storage.local.set({
       saveType: this.$data.saveTypes.findIndex(el =>
       { return el.value===app.$data.saveType.value })
  });
 },
selectCounter(event){
  this.$data.constcountDownNumber = event.target.value;
      this.openSettingItem = '';

  chrome.storage.local.set({
       constcountDownNumber:
       this.$data.constcountDownNumber
  });
},
selectResolution(item) {
      this.$data.cameraResolution = item;
      this.openSettingItem = '';

  chrome.storage.local.set({
       cameraResolution:
       this.$data.cameraResolutions.findIndex(el => {
       return el.label === app.$data.cameraResolution.label
       })
  });
  },
selectFrames(item) {
      this.$data.videoframeRate = item;
      this.openSettingItem = '';

      chrome.storage.local.set({
          videoframeRate:
          this.$data.videoframeRates.findIndex(el => {
          return el===app.$data.videoframeRate })
```

```
          });
          },

      setDisableRecord() {
      setTimeout(() => {
        if (this.selectedRecordOption.value === 'camera') {
          this.disableRecord = cameraPreview.isClosed;
        }
      });
    },
cancelShare() {
    this.requestSharePending = false;
    screenShare.cancelShare();
 },
 setConfig() {
    chrome.storage.local.set({
      config: {
        recordMode: this.$data.selectedRecordOption.value,
        }
      });
    },
 onDocumentClick() {
    this.visibleSoundSelector = false;
    this.openSettingItem = '';
 },
 getAllDevices() {
    bgPage.getAllAudioVideoDevices(devices => {
      bgPage.devicesList = devices;

      this.audioInputDevices = devices.audioInputDevices;
      this.audioOutputDevices = devices.audioOutputDevices;
      this.videoInputDevices = devices.videoInputDevices;

      if (!this.selectedAudioInput) {
        this.selectedAudioInput = this.
        audioInputDevices[0];

        chrome.storage.local.get(['selectedAudioInput'],
        function(store) {
           if (store.selectedAudioInput) {
             console.log("Input "+store.
             selectedAudioInput);
             app.$data.selectedAudioInput = app.$data.
             audioInputDevices[store.selectedAudioInput];
             console.log("Input2 "+app.$data.
             selectedAudioInput);
         }
     });
        }
        if (!this.selectedVideoInput) {
          this.selectedVideoInput =
          this.videoInputDevices[0];

       chrome.storage.local.get(['selectedVideoInput'],
       function(store) {
          if (store.selectedVideoInput) {
               app.$data.selectedVideoInput =
               app.$data.videoInputDevices
               [store.selectedVideoInput];
```

```
          }
      });
        }

        if (this.audioInputDevices.length === 0) {
          this.sound = false;
        }
      });
    },
    notification(options) {
      const defaultOptions = {
        type: 'basic',
        title: chrome.i18n.getMessage('name'),
        iconUrl: '../img/logo.png',
      };
      options = Object.assign(defaultOptions, options);
      chrome.notifications.create('', options);
    }
  }
});
```

ДОДАТОК В

HTML код вікна плагіну

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
<title></title>
<link rel="stylesheet" href="css/popup.css">
</head>
<body>
  <div id="app" @click="onDocumentClick">
  <div class="main-panel" v-if="recordStatus === 'ready'">
  <div class="header">
  <i class="setting-btn" @click="openSetting = true"></i>
  </div>
  <div class="record-options">
       <div class="record-item" v-for="item in
       recordOptions"
       :class="{active: selectedRecordOption.value ===
       item.value && !disableRecord}"
  @click="selectOption(item)">
  <i class="record-item-icon"></i>
  <p class="record-item-label">{{item.label}}</p>
  </div>
  </div>
  <div>
  <div class="switch-bar">
  <div class="switch-left">
  <span class="sound-label" @ click.stop=
  "toggleSoundSelector">
       <span class="sound-label-text">
       {{selectedSoundOption.label}}</span>
       <span class="sound-drop-icon" v-show=
       "selectedRecordOption.soundOptions.length > 1">
  <i class="drop-trigger" v-show="visibleSoundSelector">
  </i>
  </span>
  </span>
  <span class="audio-column" v-show="sound">
  <span :style="{height: audioColumnHeight + 'px'}"></span>
  </span>
     <div class="sound-options" v-show=
     "visibleSoundSelector">
       <div class="sound-item" :class="{active: item ===
       selectedSoundOption, disable: item.disable }" v-
       for="item in selectedRecordOption.soundOptions"
   @click="selectSound(item)">{{item.label}}</div>
  </div>
  </div>
       <div class="switch-right" :class="{active: sound}"
       @click="toggleSound"></div>
  </div>
       <div class="switch-bar clearfix" :class="{hide:
```

```
     select  edRecordOption.value === 'camera'}">
 <div class="switch-left" v-i18n="'cameraPreview'"></div>
   <div class="switch-right" :class="{active: camera}"
   @click="toggleCamera"></div>
</div>
</div>
<div>
<div class="start-btn" @click="start">
 <span class="start-btn-inline" v-i18n="'record'"></span>
      <span class="start-btn-mask" v-show="disableRecord"
     @click.stop></span>
</div>
</div>
     <div class="panel-mask" v-show="requestSharePending
     || countDownPending">
<div class="wrapper" v-show="requestSharePending">
 <p class="share-tips" v-i18n="'shareTips'"></p>
      <div class="cancel-btn" @click="cancelShare"
      v-i18n="'cancel'"></div>
</div>
<div class="wrapper count-down" v-show=
"countDownPending">
 <div class="count-down-bg"></div>
 <div class="count-down-number">{{countDownNumber}}</div>
</div>
</div>
 </div>
 <div class="setting-panel" :class="{open: openSetting}">
<div class="header">
     <span class="back-btn" v-i18n="'record'" @click=
     "openSetting = false"></span>
</div>
<div class="setting-body">

<div class="setting-item">
     <label class="setting-item-label" v-i18n=
     "'microphone'"></label>
     <p class="setting-item-tips" v-if=
     "!audioInputDevices.length" data-i18n=
     "lackDeviceTips"></p>
     <div class="select-options" v-if=
     "audioInputDevices.length" :class="{open:
     openSettingItem === 'audio'}">
     <p class="option-current" @click.stop=
          "openSettingItem === 'audio' ? openSettingItem =
     '' : openSettingItem = 'audio'"
     :title="selectedAudioInput.label">
     {{selectedAudioInput.label}}
     </p>
<ul class="options-list">
     <li class="options-item" v-for="item in audioIn
     putDevices" @click="selectAudioInput(item)"
     :class="{ active: item === selectedAudioInput }">
     {{item.label}}</li>
</ul>
</div>
</div>

<div class="setting-item">
```

```
        <label class="setting-item-label" v-i18n=
        "'camera'"></label>
        <p class="setting-item-tips" v-if=
        "!videoInputDevices.length" data-i18n=
        "lackDeviceTips"></p>
        <div class="select-options" v-if=
        "videoInputDevices.length" :class="
        {open: openSettingItem === 'video'}">
        <p class="option-current" @click.stop=
             "openSettingItem === 'video' ? openSettingItem =
        '' : openSettingItem = 'video'"
        :title="selectedVideoInput.label">
        {{selectedVideoInput.label}}
       </p>
    <ul class="options-list">
        <li class="options-item" v-for="item in
        videoInputDevices" @click="selectVideoInput(item)"
        :class="{ active: item === selectedVideoInput
        }">{{item.label}}</li>
    </ul>
     </div>
    </div>

    <div class="setting-group">
    <div class="setting-item" style="margin:0">
    <label class="setting-item-label" data-i18n=
    "Type"></label>
        <div class="select-options" style="width:7em" v-if=
        "saveTypes.length"   :class="{open:   openSettingItem
===
        'type'}">
    <p class="option-current" @click.stop=
        "openSettingItem === 'type' ? openSettingItem = '' :
         openSettingItem = 'type'"
     :title="saveType.label">
     {{saveType.label}}
     </p>
     <ul class="options-list">
        <li class="options-item" v-for="item in saveTypes"
        @click="selectSaveType(item)" :class="{ active: item
        === saveType }">{{item.label}}</li>
    </ul>
    </div>
    </div>

    <div class="setting-item" style="margin:0 3em 0 0">
        <p><label class="setting-item-label" data-i18n=
        "Counter"></label></p>
        <input class="options-item" style="margin-top: 6px;
        width:4em;" type="number" min="0" max="10"
        @input="selectCounter" v-model=
        "constcountDownNumber"/>
    </div>
    </div>

    <div class="setting-group">
    <div class="setting-item" style="margin:0">
        <label class="setting-item-label" data-i18n=
        "Resolution"></label>
```

```
            <div class="select-options" style="width:7em" v-if=
            "cameraResolutions.length" :class="{open:
            openSettingItem === 'resolution'}">
            <p class="option-current" @click.stop=
            "openSettingItem === 'resolution' ? openSettingItem
            = '' : openSettingItem = 'resolution'"
            :title="cameraResolution.label">
            {{cameraResolution.label}}
            </p>
        <ul class="options-list">
            <li class="options-item" v-for="item in
            cameraResolutions" @click="selectResolution(item)"
            :class="{ active: item === cameraResolution
            }">{{item.label}}</li>
        </ul>
        </div>
        </div>

        <div class="setting-item" style="margin:0 0 0 0;">
            <label class="setting-item-label" data-i18n=
            "Frames">
            </label>
            <div class="select-options" style="width:4em;" v-if=
            "videoframeRates.length" :class="{open:
            openSettingItem === 'frames'}">
            <p class="option-current" @click.stop=
            "openSettingItem === 'frames' ? openSettingItem = ''
            : openSettingItem = 'frames'"
            :title="videoframeRate">
            {{videoframeRate}}
            </p>
            <ul class="options-list">
            <li class="options-item" v-for="item in
            videoframeRates" @click="selectFrames(item)"
            :class="{ active: item === videoframeRate
            }">{{item}}</li>
        </ul>
        </div>
        </div>
        </div>
        </div>
        </div>
        <div class="record-panel" v-if="recordStatus !== 'ready'
        ">
        <div class="time">{{recordTimeFormat}}</div>
            <div class="pause-btn" :class="{continue:
            recordStatus === 'pause'}" @click="pause"></div>
        <div class="stop-btn" @click="stop"></div>
        </div>
        </div>
        <script src="js/vue/vue.js"></script>
        <script src="js/i18n.js"></script>
        <script src="js/popup.js"></script>
    </body>
    </html>
```