Универзитет у Београду
Електротехнички факултет

Софтверски додатак веб прегледача за ефикасније учење студената

Дипломски рад

Ментор:
Доц. др Дражен Драшковић

Кандидат:
Ивана Васић 0137/2016

Београд, децембар 2021.

Садржај

1. УВОД 1

2. СТУДЕНТСКЕ ЕКСТЕНЗИЈЕ 3

3. ОПИС СИСТЕМА 5

3.1. Инсталација 5
3.2. Функционална спецификација 6

4. ЕКСТЕНЗИЈА ВЕБ ПРЕТРАЖИВАЧА GOOGLE CHROME 10

4.1. Веб технологије 10
4.1.1. *HyperText Markup Language* 10
4.1.2. *Cascading Style Sheet* 11
4.1.3. *JavaScript* 12
4.2. Манифест датотека 13
4.3. Постављање екстензије на веб 13
4.4. Функционалности 14
4.4.1. Позадинска скрипта 15
4.4.2. Скрипта садржаја 15
4.5. Концепти 16
4.5.1. Догађај и слушалац догађаја 16
4.5.2. Прослеђивање порука 16
4.6. Могућности 17
4.6.1. Кодирање и декодирање *JSON* података 17
4.6.2. Складиштење података 18
4.6.3. *OAuth* 18
4.7. Преносивост на друге веб претраживаче 18

5. ИМПЛЕМЕНТАЦИЈА 20

5.1. Развојно окружење 20
5.2. Креирање манифеста 20
5.3. Интерфејс 21
5.4. Имплементација логике 24

6. ЗАКЉУЧАК 30

СПИСАК СКРАЋЕНИЦА 31

СПИСАК СЛИКА 32

ЛИТЕРАТУРА 33

1. Увод

Дана 29. октобра 1969. године рођен је интернет. Послата је прва порука, са рачунара у истраживачкој лабораторији на *UCLA* универзитету на рачунар на универзитету *Stanford*, преко тадашњег интернета, познатог као *ARPAnet*, који се користио као војни комуникациони алат америчке владе. [1] Др. *Vinton G. Cerf* и Др. *Robert E. Kahn* су 1973. године изумели *TCP/IP* протокол са идејом да повежу цео свет помоћу једне мреже којој ће свако имати бесплатан приступ. [2] Интернет је заиста постао глобална мрежа доступна свакоме, која данас броји 4,9 милијарде активних корисника, што је 63% светске популације. [3]

Први веб претраживач, *WorldWideWeb*, развио је британски информатичар *Tim Berners-Lee* 1990. године док је радио у Европској организацији за нуклеарна истраживања *CERN*, у Швајцарској. [4] [5] Веб претраживач (енгл. *web browser*), или прегледач, је апликативни софтвер за приступ интернету, чија сврха јесте да преузме садржај са веба и прикаже га на уређају корисника у виду текста, слика, видео записа и сличног садржаја. [6] Разне верзије веб претраживача су се појављивале од настанка првог па до сада, од којих су се само неке задржале, док су остале биле краткорочног карактера.

Данас постоји свега неколико веб претраживача за приступ интернету који се користе. Неки од најпопуларнијих су *Microsoft Internet Explorer*, *Microsoft Edge*, *Google Chrome*, *Mozilla Firefox*, *Safari* и *Opera*. Сви они имају бројне заједничке карактеристике. Једна од тих карактеристика је могућност додавања такозваних проширења.

Софтверски додатак веб претраживача, екстензија (енгл. *extension*) или проширење претраживача (у даљем тексту: екстензија) је софтверски програм, изграђен на веб технологијама, који омогућава корисницима да прилагоде претраживач према својој жељи додавањем елемената подразумеваном корисничком интерфејсу и тиме побољша искуство прегледања. [7]

Internet Explorer је био први велики претраживач који је подржавао екстензије 1999. године. [8] *Mozilla Firefox* подржава екстензије од свог објављивања 2004. године. *Opera* је почела да подржава екстензије 2009. године, а и *Google Chrome* и *Safari* су то учинили следеће године. *Microsoft Edge* је додао подршку за проширења 2016. године. [9]

У *Chrome* веб продавници (енгл. *Chrome Web Store*) је данас доступно преко 100 хиљада екстензија, што пружа широку разноврсност избора за кориснике прегледача *Chrome*. [10]

У овом раду биће приказана израда екстензије за *Google Chrome* прегледач коришћењем веб технологија као што су *HTML* (*HyperText Markup Language*), *CSS* (*Cascading Style Sheets*) и *JavaScript*.

Циљ рада је да помогне студентима да побољшају квалитет учења помоћу екстензије која нуди различите варијанте временске организације. У другом поглављу дат је преглед неких од постојећих екстензија које такође имају за циљ поспешивање продуктивности студената, уз кратак опис сваке.

У трећем поглављу биће речи о томе како је настала инспирација за овај рад и о крајњем резултату истог, и поред тога, биће дат детаљан опис коришћења система уз слике и кратка објашњења постојећих функционалности.

Екстензија је рађена за *Google Chrome* веб продавницу и конкретно је прилагођена *Google Chrome* претраживачу, па је у четвртом поглављу обрађена екстензија за *Chrome*, тачније њена структура и функционалности, као и технологије коришћене за израду. Како је израда екстензије веома слична и када су различити веб претраживачи у питању, обрађена је и тема прилагођености *Chrome* екстензија другим претраживачима и разлике екстензија различитих веб претраживача.

Опис имплементације система дат је у петом поглављу. Наведено је коришћено окружење за развој пројекта - *Visual Studio Code*. Датотеке потребне за израду екстензије, укључујући и обавезну датотеку *manifest.json* описане су уз убачене делове кода на којима се види њихов начин рада.

Шесто поглавље, уједно и последње, представља закључна разматрања која дају осврт на цео рад - оно што има за циљ и крајње резултате.

2. Студентске екстензије

Проширења за *Chrome* су одличан начин за аутоматизацију школских задатака, пошто су веома лака за инсталирање, коришћење и уграђена директно у сам прегледач. Неке од најчешће преузиманих екстензија које нуде различите олакшице и могућности студентима су наведене у следећем тексту.

StayFocusd

Привремено блокира сајтове (енгл. *website*) у *Chrome* претраживачу, како се учење не би претворило у скроловање кроз друштвене мреже, гледање видео записа или неки други вид дистракције.

Поред тога постоји и опција "*Nuclear Option*" која доводи ово до крајности тако што прекида приступ свему - осим сајтовима који се дозволе. Када се изабере ова опција, нема начина да се искључи софтвер за блокирање док се одбројавање не заврши. [11]

Todoist

Може да централизује све активности, задатке, референце и пројекте на којима студент ради у једној апликацији тако да може лако да приступи својим листама и да се осврне на свој напредак. Да би се добила боља представа о временском оквиру, може и да се синхронизује апликација са Google календаром (енгл. *Google Calendar*).

Такође, има функцију "*Karma*" која кориснику даје бодове на основу количине дневних и недељних циљева које постигне. [12]

Evernote Web Clipper

Evernote је најпопуларнији алат за дигитално вођење белешки на свету, а ово је његово проширење. При наиласку на неку објаву или игнографику - визуалну презентацију информација, података или знања, коју корисник сматра значајном, има могућност да је сачува. Све што треба да уради је да кликне на "*Clip*" да би снимио белешку са веб странице, е-поште или било ког другог документа на централну локацију.

Evernote такође има мноштво алата за означавање (енгл. *annotating*) тако да је могуће правити белешке директно на сачуваним клиповима, користећи текст да би се скренула пажња на најважније делове слика. [13]

Google Dictionary

Када ученици наиђу на реч коју не разумеју на мрежи, са *Google* речником, могу само двапут да кликну на реч и појавиће се искачући облачић са дефиницијом. То штеди време потребно за отварање још једне картице да би се добила дефиниција. Такође подржава језике изван енглеског, укључујући француски, немачки, италијански и шпански. [14]

Nimbus

Омогућава сликање и снимање екрана претраживача. Омогућава снимање екрана целе странице, као и снимање само одабраног дела странице. Такође омогућава кориснику да коментарише и уређује снимке екрана директно у самом прегледачу. Такође, корисник може додати свој водени жиг (енгл. *watermark branding*) на све снимке екрана са само неколико кликова.

Могуће је додавање текста и слика на снимке екрана. Такође могуће је замутити делове слика са само неколико кликова. *Nimbus* може помоћи кориснику да ухвати важне информације управо онако како су приказане на страници. [15]

RescueTime

Одлична апликација за *iOS* и *Android*, говори ученицима тачно колико времена проводе у активној картици (енгл. *tab*) или прозору *Chrome* претраживача или *ChromeOS* уређаја. Категорише сајтове које посећују и оцењује их од веома продуктивних до веома ометајућих. Прати шта корисник ради и блокира сајтове којима не жели да приступи, тако да му не ометају пажњу. [16]

Weava Highlighter

Проширење за *Chrome* које омогућава кориснику да обележава текст (енгл. *highlighting*) – PDF документе и веб странице различитим стиловима и бојама по избору. *Weava* је радни простор за истраживања и студије, који приказује обележене делове текста и друге функције на активном *URL* веб-сајта и *PDF* документима отвореним у прегледачу. Омогућава кориснику да креира цитат, организује своје обележене ставке и приступи својим белешкама у било ком тренутку. [17]

3. Опис система

Учење је један од најзахтевнијих задатака са којима се треба суочити као студент. Према бројним глобалним истраживањима, просечно време учења једног студента је најмање четири до пет сати свакога дана. Поред осталих обавеза са којима се студенти сусрећу, некада је тешко или немогуће испунити споменути број сати, па због тога често долази до одлагања учења и гомилања градива.

Проблем данас је тај што многи људи верују да морају да раде дуже да би урадили више. Како се испоставило, недостатак дисциплине или мотивације није разлог кратког трајања пажње. Разлог за то је еволуциона биологија - људски мозак не може дуго да се фокусира на једну исту радњу.

Истраживање Алехандра Љераса (шпан. *Alejandro Lleras*), са Универзитета у Илиноису (енгл. *Illinois*), показало је да прекидање и поновно започињање рада омогућава да останемо фокусирани. [18] Када се обављају дуги задаци, као што су учење за испите, прављење презентација или писање извештаја, препоручљиво је да се праве кратке и планиране паузе. Приликом континуалног дугог рада, лако је изгубити концентрацију и направити грешку, која касније може само да проузрокује потребне додатне сате рада. Прављење стратешких пауза повећава квалитет учења.

Циљ овог рада је управо да омогући студентима да на једноставан начин управљају својим временом, буду ефикасни, и тиме побољшају квалитет самог учења помоћу екстензије која нуди различите варијанте временске организације.

Како је *Google Chrome* тренутно водећи интернет претраживач на свету са глобалним тржишним уделом од 69,28%, екстензија је прилагођена првенствено њему. [19]

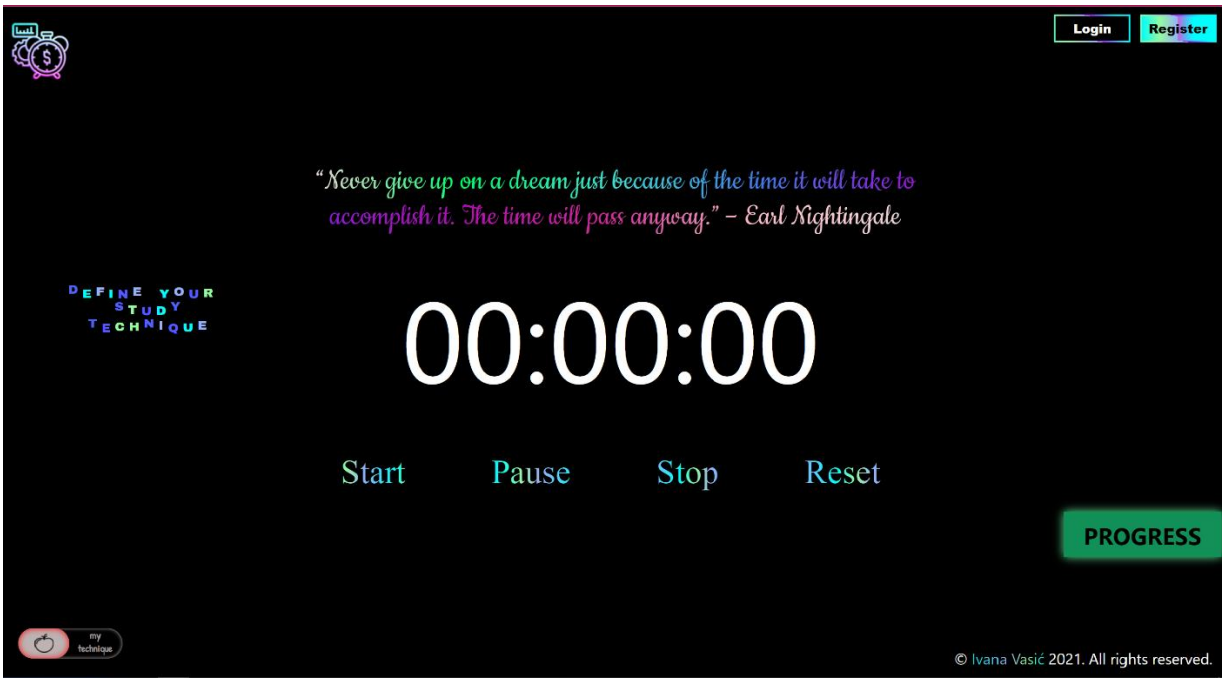
3.1. Инсталација

Процес инсталације екстензије је веома једноставан и изводи се у свега неколико корака. Након отварања веб претраживача *Google Chrome*, треба ући у *Chrome Web Store*, веб-продавницу, и ту пронаћи жељену екстензију, са могућством претраге по називу екстензије. И последњи преостали корак јесте додавање те екстензије кликом на дугме "*Add to Chrome*" и затим потврдом те акције кликом на дугме "*Add extension*". Потом што

је додата, екстензији се може приступити преко иконице на десној страни адресне линије (енгл. *address bar*). [20]

3.2. Функционална спецификација

Главни део екстензије је бројач времена (у даљем тексту: тајмер) који служи за мерење појединачних временских интервала. Идеја је да студент учи у интервалима раздвојеним кратким паузама, како би одржао концентрацију.

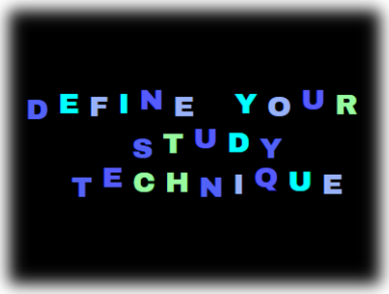


Слика 1. Приказ екстензије

Екстензија пружа мотивацију у виду инспиративних и охрабрујућих цитата, који се насумично мењају приликом сваког поновног отварања, као и на сваких 30 секунди одбројаног времена.

Како је циљ ове екстензије да омогући кориснику да сам организује време онако како њему највише одговара, постоји могућност да одабере количину трајања интервала рада и трајања паузе. На страници екстензије налази се опција "*Define your study technique*" помоћу које то може да уради. Преласком курсора преко датог текста, отвара се прозор у коме могу да се дефинишу наведени временски интервали у минутима и сачувају кликом на дугме "*Save*". Уколико корисник не дефинише дате интервале, биће

узети подразумевани, а то су: 60 минута рада и 15 минута паузе. Такође је могуће дефинисати и само један од интервала, док ће овај други имати подразумевано време.



Слика 2. Опција дефинисања технике

Слика 3. Дефинисање технике

Постоји и друга опција фиксних интервала, а то су: 25 минута рада и 5 минута паузе. Реч је о познатој "*Pomodoro*" техници - метода управљања временом коју је развио *Francesco Cirillo* касних 1980. инспирисан кухињским тајмером који ради по том принципу, који је користио као студент. Активира се дугметом на коме је иконица парадајза (итал. *pomodoro*). [21]

Слика 4. Pomodoro укључен

Слика 5. Pomodoro искључен

Контролна дугмад (енгл. *button*) истакнута су одмах испод тајмера и нуде опције започињања мерења (енгл. *start*), паузирања (енгл. *pause*), заустављања (енгл. *stop*) и ресетовања (енгл. *reset*) времена. Притиском на дугме "*Start*", време почиње да се одбројава и одбројано време од почетка рада се види на тајмеру и ажурира сваком секундом која прође. Уколико студент није почео или пак одустане од мерења, може да поништи претходно започето одбројавање тиме што ће ресетовати тајмер на почетак - на нулу, притиском на дугме "*Reset*". Када одређена спољашња активност прекине корисника у току фиксног интервала рада, он има могућност да паузира тренутни интервал притиском на дугме "*Pause*" и настави чим заврши са тим што га је прекинуло, поновним притиском на дугме "*Start*". Уколико корисник ипак одлучи да заврши са својим радом док је интервал рада још у трајању, у могућности је да прекине свој рад

притиском на дугме “*Stop*”. Разлика између заустављања и рестартовања тајмера је у чувању до тада искоришћеног времена за учење, о чему ће речи бити ускоро.

Након што се одброји задато или подразумевано време у одређеном режиму (корисничка или *pomodoro* техника) зазвони аларм, који обавештава корисника да је истекло потребно време за рад или паузу, што је истакнуто и на самом екрану - о ком интервалу је реч. Тада се чека потврда корисника за наставак, то јест за започињање наредног временском интервала.

Слика 6. Интервал рада

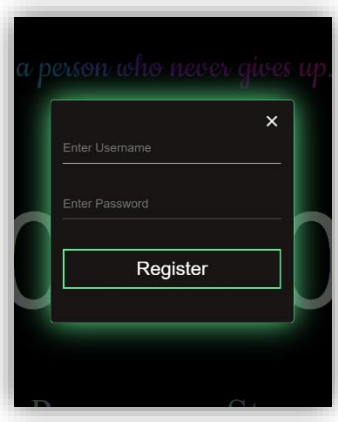
Слика 7. Интервал паузе

Истакнут је и напредак корисника на страници у виду мале књижице коју корисник отвара преласком миша преко ње и унутра може да види свој дневни прогрес то јест колико је укупно времена учио тог дана, који се ажурира при свакој новој секунди рада.

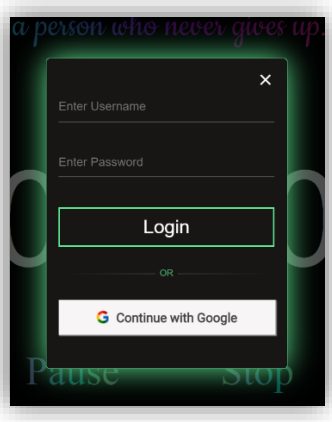
Слика 8. Прогрес књижица

Слика 9. Приказ прогреса

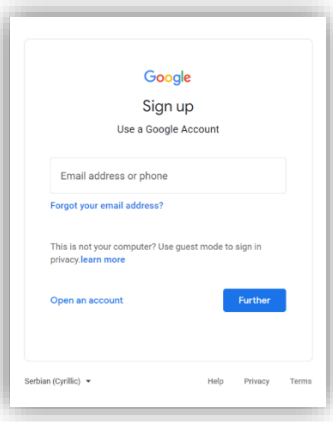
Корисник има могућност да направи свој налог и да се улогује у систем. Могуће је и улоговати се помоћу e*mail* адресе која се дохвата са тренутно активног *Google* налога или, уколико не постоји активан *Google* налог, пали се прозор који омогућава кориснику да се улогује на *Google* налог, а затим се аутоматски, након што се улогује, дохвата *mail* адреса и приступа налогу на екстензији везаном за ту адресу.



Слика 10. Register форма



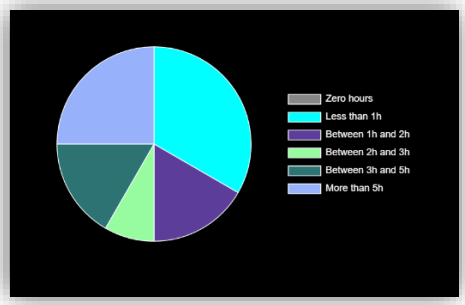
Слика 11. Login форма



Слика 12. Google login

Након успешне аутентикације при логовању у систем или прављења налога, појављује се једна додатна карактеристика, а то је историја учења. Кориснику се на страници одмах испод тајмера приказује графикон са његовим укупним прогресом. Графикон се ажурира сваки следећи пут када је корисник активан, то јест када учи. Циљ је да се евидентира колико студент учи на дневном нивоу и направи процентуална статистика тих података помоћу специјалног графикона (енгл. *chart*).

Слика 13. Улогован режим



Слика 14. Графикон учења

Кликом на иконицу корисника у горњем десном углу отвара се кориснички мени. Корисник има могућност да се излогује из система, а сви подаци ће остати заувек сачувани, уколико корисник не одлучи да их избрише опцијом "*Clear history*" у корисничком менију.

4. Екстензија веб претраживача Google Chrome

Софтверски додатак веб претраживача, екстензија или проширење претраживача је софтверски програм, изграђен на веб технологијама, који омогућава корисницима да прилагоде претраживач према својој жељи додавањем елемената подразумеваном корисничком интерфејсу и тиме побољша искуство прегледања.

Google Chrome екстензије су апликације које се покрећу унутар *Chrome* претраживача и пружају додатне функционалности, интеграцију са веб адресама или услугама трећих страна и прилагођено (енгл. *customized*) искуство прегледања.

Екстензије су направљене од различитих, али међусобно кохерентних компоненти. Компоненте могу укључивати позадинске скрипте (енгл. *background scripts*), скрипте садржаја (енгл. *content scripts*), страницу са опцијама (енгл. *options page*), елементе корисничког интерфејса и разне логичке датотеке. Компоненте екстензије се креирају помоћу технологија за веб развој: *HTML, CSS* и *JavaScript*. [7]

4.1. Веб технологије

Екстензије су написане коришћењем истих стандардних веб технологија које програмери користе за креирање веб-сајтова. *HTML* се користи као језик за означавање садржаја, *CSS* се користи за стилизовање, а *JavaScript* за скриптовање. Пошто *Chrome* подржава *HTML5* и *CSS3*, програмери могу да користе најновије отворене веб технологије као што су платно (енгл. *canvas*) и *CSS* анимације у својим екстензијама. *JavaScript API,* који помажу у обављању функција попут *JSON* кодирања и интеракције са прегледачем, доступни су за приступ екстензијама.

За развој овог софтверског програма користе се поменуте веб технологије о којима ће бити мало више речи у наредним потпоглављима.

4.1.1. *HyperText Markup Language*

HyperText Markup Language, или скраћено *HTML*, је најосновнији градивни блок веба. Служи за креирање веб страница, дефинише значење и структуру веб страница. "*Hypertext*" се односи на везе (енгл. *link*) које повезују веб странице једне са другима, било у оквиру једног веб-сајта или између веб-сајтова. Везе су основни аспект веба.

Учитавањем садржаја на Интернет и повезивањем са страницама које су креирали други људи, постаје се активан учесник у *World Wide Web* свету.

HTML користи "*markup*" језик. "*Markup*" језик је компјутерски језик који користи ознаке за дефинисање елемената унутар документа - означавање текста, слика и другог садржаја за приказ у веб претраживачу. Читљив је људима, што значи да датотеке за означавање садрже стандардне речи, а не типичну програмску синтаксу. *HTML* ознаке укључују посебне елементе као што су *<head>*, *<title>*, *<body>*, *<header>*, *<footer>*, *<p>*, * *, **, *<img>*, *<ul>*, *<ol>*, *<li>* и многи други.

HTML се састоји од низа елемената, који говоре претраживачу како да прикаже садржај. Елемент је одвојен од другог текста у документу помоћу тагова (енгл. *tag*), који се састоје од назива елемента окруженог са "<" и ">". Име елемента унутар ознаке не разликује велика и мала слова. То јест, може се писати великим, малим словима или мешано. На пример, ознака *<head>* може бити написана као *<Head>*, *<HEAD>* или на било који други начин. [22]

4.1.2. *Cascading Style Sheet*

Cascading Style Sheet, или скраћено *CSS*, се користи за форматирање изгледа веб страница. Може се користити за дефинисање стилова текста, величина табела и других аспеката веб страница који су раније могли бити дефинисани само у оквиру *HTML* датотеке странице. Првобитно, *HTML* је служио да дефинише комплетан изглед, структуру и садржај веб-странице, али је од *HTML* верзије 4.0 уведен *CSS* који би дефинисао конкретан изглед, док је *HTML* остао у функцији дефинисања структуре и садржаја.

CSS помаже веб програмерима да креирају уједначен изглед на неколико страница веб-сајта. Често коришћени стилови морају бити дефинисани само једном у *CSS* документу. Једном када је стил дефинисан у *CSS* датотеци, може га користити било која страница која укључује ту *CSS* датотеку. Осим тога, *CSS* олакшава промену стилова на неколико страница одједном.

Иако је *CSS* одличан за креирање стилова текста, користан је и за форматирање других аспеката изгледа веб странице. На пример, *CSS* се може користити за дефинисање допуна ћелија у ћелијама табеле, стила, дебљине и ивице табеле, као и допуна око слика

или других објеката. *CSS* даје веб програмерима прецизнију контролу над изгледом веб страница него *HTML*. Због тога већина веб страница данас укључује *CSS*. [23]

4.1.3. *JavaScript*

JavaScript, или скраћено *JS*, је динамичан, слабо типизиран и интерпретиран програмски језик високог нивоа. Иако је најпознатији као скриптни језик за веб странице, користе га и многа окружења која нису претраживачи, као што су *Node.js*, *Apache CouchDB* и *Adobe Acrobat*.

Стандарди за *JavaScript* су *ECMAScript Language Specification* (*ECMA-262*) и *ECMAScript Internationalization API specification* (*ECMA-402*). *JavaScript* је једна од три водеће технологије за дефинисање садржаја на вебу, поред веб технологија *HTML* и *CSS*. Већина веб-сајтова користи *JS*, а сви модерни веб-читачи га подржавају без потребе за инсталирањем додатака. У комбинацији са *HTML* језиком и *CSS* језиком *JavaScript* чини *DHTML* (*Dynamic HTML*).

JS је базиран на прототиповима, вишепарадигматски, једнонитни, динамички језик, који подржава објектно-оријентисане, императивне и декларативне (нпр. функционално програмирање) стилове. Садржи *API* (енгл. *Application Programming Interface* - програмски интерфејс апликације) за рад са текстом, низовима, датумима и регуларним изразима, али не и улазно/излазне функционалности, као што су повезивање, складиштење података или графичке функционалности, за шта се ослања на окружење у коме се извршава.

JavaScript је најпопуларнији скриптни језик на Интернету којег подржавају сви познатији прегледачи (*Internet Explorer, Chrome, Mozilla Firefox, Opera, Safari*). Неке од примена овог језика су:

- Може динамички да уноси код у *HTML* страницу,
- Може да реагује на догађаје - може да се подеси тако да се изврши кад се нешто деси, нпр. кад се страна учита или кад корисник кликне на *HTML* елемент,
- Може да прочита или испише *HTML* елементе,
- Може да прочита и да промени садржај *HTML* елемента,
- Може да се користи за проверу исправности унетих података (у форму, пре него што се пошаљу на сервер),
- Може да се користи за детектовање корисника претраживача - у зависности од претраживача, учитава се страна специјално дизајнирана за тај претраживач,

- Може да се користи за креирање колачића (енгл. *cookies*),
- Може да се користи за чување и враћање информација о рачунару посетиоца, итд. [24]

4.2. Манифест датотека

Датотека *manifest.json* је једноставна *JSON* (*JavaScript Object Notation*) датотека на веб адреси корисника која прегледачу говори о корисничкој веб адреси на мобилном уређају или десктопу корисника. Претраживач *Chrome* захтева да има ову датотеку да би се приказао упит "*Add to Home Screen*".

Када корисник инсталира или истакне (енгл. *bookmark*) веб апликацију на почетни екран или је дода у покретач апликација (енгл. *application launcher*), *manifest.json* пружа претраживачу да може да види веб-сајт са именом, иконама итд. Затим постоје напредније функције, као што је могућност да се означи жељена оријентација и да ли програмер жели да апликација буде преко целог екрана.

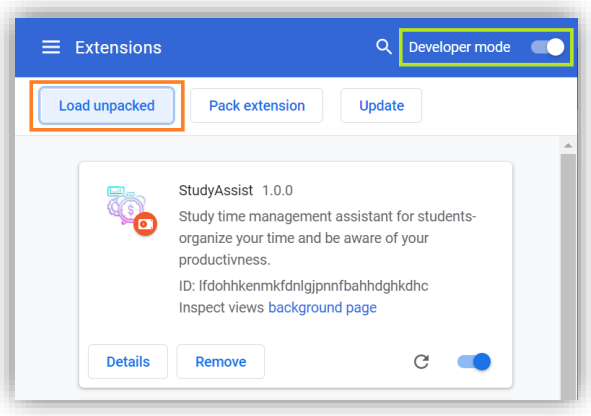
Ова датотека садржи детаље о називу екстензије, опису, аутору, иконе које треба да користи, почетни *URL* (енгл. *Uniform Resource Locator*), то јест веб-адреса, на којој треба да се појави екстензија када се покрене, и многе друге детаље о екстензији о којима ће тек бити речи у наредном тексту. [20]

4.3. Постављање екстензије на веб

Директоријум који садржи датотеку *manifest.json* може се додати као екстензија у режиму програмера у свом тренутном стању. Кораци су следећи:

1. Отворити страницу за управљање екстензијама на неки од начина:
    - отићи на веб адресу *chrome://extensions* или
    - кликнути на иконицу пузле у горњем десном углу адресног бара и затим одабрати опцију "*Manage extensions*" или
    - отворити "*Tools*" мени кликом на иконицу са натписом (енгл. *tooltip*) "*Customize and control Google Chrome*" у склопу адресног бара скроз десно, а затим у опцији "*More tools*" изабрати "*Extensions*".
2. Омогућити режим програмера помоћу клика на прекидач поред режима програмера са натписом "*Developer mode*".

3. Кликнути на дугме “*Load unpacked*” које отвара прозор за одабир директоријума екстензије са свим њеним фајловима, и учитава је тако распаковану у виду директоријума, без потребе за компресовањем.



Слика 15. Учитавање екстензије

Када су испоштовани сви кораци, екстензија је успешно инсталирана и спремна за даљу надоградњу, а сада и уз могућност провере у самом веб претраживачу. [20]

4.4. Функционалности

Након што је екстензија инсталирана, не ради ништа док јој се не каже шта тачно да ради. Ту наступају друге важне датотеке које одређују рад саме екстензије. Те датотеке су *background.js* и *content.js* скрипте. Поред њих, постоји *HTML* скрипта која представља садржај екстензије и *CSS* скрипта која даје изглед екстензији, оне су специфичне за сваку апликацију, и може бити и неколико *HTML* и *CSS* датотека са наведеном улогом.

4.4.1. Позадинска скрипта

Позадинске скрипте (енгл. *background script*) се региструју у манифесту под пољем "*background*". Регистровање позадинске скрипте у манифесту говори екстензији на коју датотеку треба да се позива и како та датотека треба да се понаша.

Chrome је сада свестан да проширење укључује позадинску скрипту. Када екстензија поново буде учитана, *Chrome* ће скенирати наведену датотеку за додатна упутства, као што су важни догађаји (енгл. *event*) које треба да слуша.

Кључ "*persistent*" треба навести као нетачан. Једина прилика да позадинска скрипта остане стално активна је ако екстензија користи *chrome.webRequest API* за блокирање или измену мрежних захтева. *WebRequest API* није компатибилан са непостојаним позадинским страницама.

Више позадинских скрипти се може регистровати за модуларни код. Синтаксно, називи скрипти у облику "*backgroundScriptName.js*" би се додавали, раздвојени зарезима, у подпоље "*scripts*" поља "*background*".

Структурирање позадинске скрипте базира се на догађајима од којих екстензија зависи. Дефинисање функционално релевантних догађаја омогућава позадинским скриптама да мирују док се ти догађаји не покрену и спречава екстензију да пропусти важне окидаче. [25]

4.4.2. Скрипта садржаја

Скрипте садржаја (енгл. *content script*) су датотеке које се покрећу у контексту веб страница. Коришћењем стандардног модела објеката документа (енгл. *Document Object Model - DOM*), они су у могућности да читају детаље веб страница које прегледач посећује, уносе промене у њих и прослеђују информације својој надређеној екстензији.

Скрипте садржаја могу да приступе интерфејсима *Chrome API* које користи њихова надређена екстензија разменом порука са екстензијом. Они такође могу приступити адреси *URL* датотеке екстензије помоћу *chrome.runtime.getURL()* и користити резултат на исти начин као и неки други *URL*.

Скрипте садржаја постоје у изолованом свету, дозвољавајући скрипти садржаја да изврши промене у свом *JavaScript* окружењу без сукоба са страницом или другим скриптама садржаја екстензије.

Скрипте садржаја могу бити декларисане статички или програмски убачене.

Декларације скрипте статичког садржаја у *manifest.json* користе се за скрипте које треба аутоматски да се покрећу на добро познатом скупу страница. Статички декларисане скрипте се региструју у манифесту у пољу "*content_scripts*". Оне могу укључивати *JavaScript* датотеке, *CSS* датотеке или обоје. Све скрипте садржаја за аутоматско покретање морају специфицирати обрасце подударања (енгл. *match patterns*).

Програмско убацивање скрипти садржаја користи се за скрипте које треба да се покрећу као одговор на догађаје или у одређеним приликама.

Иако су окружења за извршавање скрипти садржаја и странице које их хостују (енгл. *host* - домаћин) изоловане једна од друге, деле приступ моделу објеката документа странице. Ако страница жели да комуницира са скриптом садржаја или са екстензијом преко скрипте садржаја, то мора учинити преко дељеног модела објеката документа. Пример се може постићи коришћењем *window.postMessage*. [26]

4.5. Концепти

4.5.1. Догађај и слушалац догађаја

Догађај (енгл. *event*) је радња или појава коју апликација може идентификовати и која има значај за апликацију и њену намену.

Слушаоци догађаја (енгл. *event listener*) морају бити регистровани синхроно од почетка странице. Слушалац догађаја *runtime.onInstalled* служи да би се иницијализовала екстензија приликом инсталације. Овај догађај се користи за постављање стања или за једнократну иницијализацију. Дефинише се увек у оквиру позадинске скрипте. Слушаоци се не смеју регистровати асинхроно, јер се неће правилно покренути. Слушаоци служе да позову функцију када се догађај деси. Унутар слушаоца догађаја имплементира се жељена реакција на догађај. [27]

4.5.2. Прослеђивање порука

Пошто се скрипте садржаја покрећу у контексту веб странице, а не екстензије, често им је потребан неки начин комуникације са остатком екстензије. Комуникација између екстензија и њихових скрипти садржаја функционише коришћењем прослеђивања порука. Свака страна може да слуша поруке послате са друге стране и да одговори на истом каналу. Порука може да садржи било који важећи *JSON* објекат (*null*, логички, број, стринг, низ или објекат).

Постоји једноставан *API* за једнократне захтеве и сложенији *API* који омогућава дуготрајне везе за размену више порука са заједничким контекстом. Такође је могуће послати поруку другој екстензији ако је познат њен идентификациони број.

Ако се шаље само једна порука другом делу екстензије (и опционо одговара на ту поруку), требало би да се користи поједностављени *runtime.sendMessage* или *tabs.sendMessage*. Ово омогућава слање једнократне *JSON* поруке из скрипте садржаја у екстензију, или обрнуто. Опциони параметар повратног позива омогућава да управљање

одговором са друге стране. Одговор, ако постоји, шаље се као параметар функције прослеђене при слању поруке.
Слање захтева из екстензије у скрипту садржаја изгледа веома слично, осим што треба навести на коју картицу се шаље.

На пријемној страни, потребно је подесити слушалац догађаја *runtime.onMessage* да обрађује поруку. Ово изгледа исто са скрипте садржаја или странице екстензије.

Ако се асинхроно шаље одговор на поруку, додаје се *return true;* у обрађивач (енгл. *handler*) догађаја *onMessage*. Ако више страница ослушкује *onMessage* догађаје, само ће прва која позове функцију за слање одговора на одређени догађај успети да пошаље одговор. Сви други одговори на тај догађај биће игнорисани. [28]

4.6. Могућности

Екстензије имају безброј могућности, овде ће бити издвојене неке од њих, између осталог и оне које су коришћене у овом раду.

4.6.1. Кодирање и декодирање *JSON* података

Модерне *JavaScript* машине (енгл. *engine*) као што је *Chrome B8* имају уграђену подршку за *JSON.stringify* и *JSON.parse* тако да је могуће користити ове функције у екстензијама без укључивања *JSON* библиотека у код. Метода *JSON.stringify* конвертује *JavaScript* објекат или вредност у *JSON* стринг (енгл. *string*) - низ знакова, опционо замењујући вредности ако је наведена функција замене или опционо укључујући само наведена својства ако је наведен низ замене. Метода *JSON.parse* рашчлањује (енгл. *parse*) *JSON* стринг, конструишући *JavaScript* вредност или објекат описан стрингом. [29] [30]

4.6.2. Складиштење података

Екстензије могу да користе локално складиште - *localStorage* за трајно складиштење података у виду стринга (тип података који се користи у програмирању за представљање текста). Потребно је декларисати дозволу за складиштење у манифесту екстензије да би се користио *API* за складиштење.

Користећи *JSON* функције уграђене у *Chrome*, могу се складиштити сложене структуре података у *localStorage*. За проширења која треба да изврше *SQL* упите над својим ускладиштеним подацима, *Chrome* имплементира *SQL* базе података на страни

клијента, које се такође могу користити. Екстензије могу да складиште до пет мегабајта података у локалном складишту. [29]

4.6.3. *OAuth*

Постоје екстензије које користе *OAuth* за приступ програмским интерфејсима апликација удаљених података. Већина програмера сматра да је користи *JavaScript OAuth* библиотеку како би се поједноставио процес потписивања *OAuth* захтева. Више о томе у петом поглављу. [29]

4.7. Преносивост на друге веб претраживаче

Првобитно, већина претраживача се ослањала на сопствени јединствени код. Последњих година је све више претраживача почело да ради из заједничког, отвореног (енгл. *open-source*) *Chromium* изворног кода, који покреће *Google Chrome*, као и *Edge* и *Brave*. Пошто су сви ови прегледачи засновани на изворном коду *Chromium*, сви раде са *Chrome* екстензијама.

Екстензије направљене помоћу *WebExtension API* су дизајниране да буду компатибилне са екстензијама за *Chrome* и *Opera* претраживаче. Екстензије написане за те претраживаче треба да раде на *Firefox* претраживачу са минималним променама. Међутим, постоје значајне разлике између *Chrome, Firefox* и *Edge* претраживача:

- Подршка за *JavaScript API* се разликује у различитим прегледачима.
- Подршка за *manifest.json* кључеве се разликује у различитим прегледачима.
- Приступ према *Javascript API*:
  - *Firefox* и *Edge*: Приступ према *JavaScript API* је под именским простором "*browser*".
  - Chrome: Приступ према *JavaScript API* је под именским простором "*chrome*"
- Асинхрони *API*:
  - *Firefox*: Асинхрони *API* се имплементира помоћу обећања (енгл. *promises*).
  - *Chrome* и *Edge*: Асинхрони *API* се имплементира помоћу повратних позива. (енгл. *callback*).

Firefox подржава и "*chrome*" и "*browser*" именске просторе. Као помоћ при преносу, *Firefox* имплементација *WebExtensions* подржава "*chrome*", користећи повратне позиве, као и "*browser*", користећи обећања. То значи да ће многа проширења за *Chrome* радити у *Firefox* претраживачу без икаквих промена. [31]

5. Имплементација

5.1. Развојно окружење

Програм коришћен за израду екстензије је *Visual Studio Code*, или скраћено *VS Code*. *Visual Studio Code* је поједностављен уређивач (енгл. *editor*) кода, који је направио *Microsoft* за *Windows*, *Linux* и *macOS*, са подршком за операције развоја као што су отклањање грешака (енгл. *debugging*), покретање задатака (енгл. *task running*) и контролу верзија (енгл. *version control*). *VS Code* се може користити са различитим програмским језицима, укључујући и *JavaScript*. Испоручује се са алатом за допуну кода *IntelliSense* за *JavaScript*, *JSON*, *CSS*, и *HTML*.

Први корак је креирање директоријума у коме ће бити смештене све датотеке пројекта и отварање тог директоријума у радном простору (енгл. *workspace*) окружења *VS Code*. [32]

5.2. Креирање манифеста

Датотека манифест је потребна за сва проширења за *Chrome*. Манифест датотека говори *Chrome* претраживачу све што треба да зна да би правилно учитао екстензију. Стога, наредни обавезни корак је креирање датотеке ***manifest.json*** у оквиру првобитно креираног фолдера.

```
{
  "name": "StudyAssist",
  "version": "1.0.0",
  "description": "Study time management assistant for students- organize
your time and be aware of your productivness.",
  "manifest_version": 2,
  "icons": {
    "48": "./images/time-icon.png"
  },
  "background": {
    "scripts": ["background.js"],
    "persistent": false
  },
  "permissions": [
    "tabs",
    "idle",
    "identity",
    "identity.email",
```

```
    "storage",
    "activeTab",
    "webNavigation"
  ],
  "oauth2": {
    "client_id": "154483090163-
joiepoq9j1mbede019eap3m8tcfaju8f.apps.googleusercontent.com",
    "scopes": ["profile email", "https://www.googleapis.com/auth/contacts"]
  },
  "content_security_policy": "script-src 'self' 'unsafe-eval'
https://cdn.jsdelivr.net https://rawgit.com; object-src 'self'",
  "web_accessible_resources": [
    "./sounds/alarm-sound.mp3"
  ]
}
```

Делимично је приказан код манифест датотеке који, поред имена, верзије и описа екстензије, верзије манифеста и иконице која представља лого екстензије, садржи дефиниције и дозволе потребне за жељени рад екстензије. Помоћу поља "*permissions*" дозвољавају се разне акције при употреби екстензије, неке од оних које су коришћене у овом пројекту су:

- "*tabs*" - Даје екстензији приступ привилегованим пољима *Tab* објеката које користи неколико *API*, укључујући *chrome.tabs* и *chrome.windows*,
- "*idle*" - Користи *chrome.idle API* да открије када се стање мировања машине променило,
- "*identity*", "*identity.email*" - Улога у приступању информацијама везаним за аутентикацију на самом претраживачу,
- "*storage*" - Омогућава коришћење локалног складишта,
- "*activeTab*" - Даје дозволу за приступ тренутно активној картици,
- "*webNavigation*" - Користи *chrome.webNavigation API* за примање обавештења о статусу захтева за навигацију на вебу.

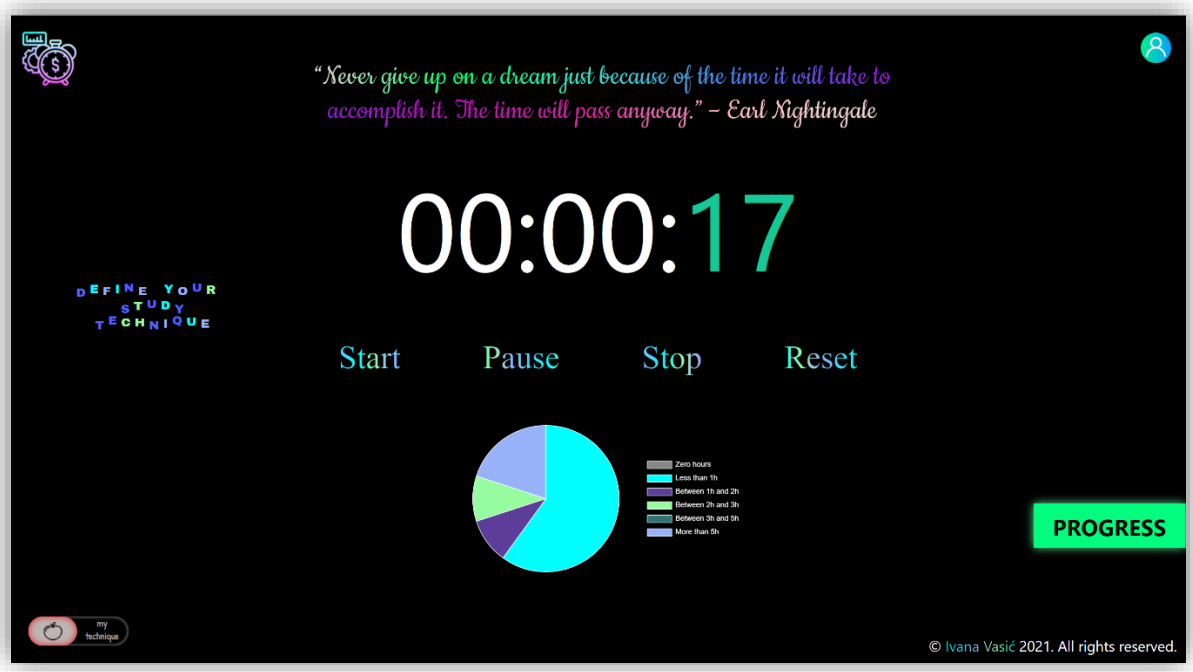
Остале функционалности биће објашњене у даљем тексту - када и зашто су додате у манифест датотеку, као и како доприносе раду саме екстензије.

Када је манифест датотека написана, екстензија се може отпремити на интернет на начин описан у 4.3. поглављу. [24]

5.3. Интерфејс

Након што је екстензија инсталирана, појављује се иконица у горњем десном углу на адресној линији са логом екстензије дефинисаним у манифесту. Кликом на ту иконицу, екстензија се отвара. Екстензије се могу отварати на неколико начина, на пример у виду искачућег блока (енгл. *popup*) или нове картице, а неке само започињу свој рад у позадини као реакција на клик на иконицу екстензије.

У овом раду, екстензија се отвара у виду нове картице у претраживачу. Ради се о страници која представља екстензију и са којом корисник интерагује. Страница се ради у већ поменутој *HTML* технологији, помоћу које дефинишемо распоред и организацију компонената на страници. Датотека која представља ту страницу ***content.html*** смешта се у исти фолдер као и манифест датотека. Поред ње додају се још две датотеке помоћу којих је стилизована страница, то су ***login-style.css*** и ***style.css***. Ове датоте рађене су поменутом *CSS* технологијом и дају изглед страници.



Слика 16. Интерфејс екстензије

Контроле за управљање тајмером су на располагању кориснику у виду дугмића "*Start*", "*Stop*", "*Pause*" и "*Reset*" одмах испод приказаног времена тајмера. Рад тајмера

може бити различит и корисник има могућност одабира жељеног рада. Одабир се врши на два начина:

- помоћу дугмета у виду прекидача у левом углу странице, које омогућава кориснику да се определи за већ дефинисану технику учења "*Pomodoro*", тиме што ће укључити прекидач, или
- помоћу натписа "*Define your study technique*" на левом делу странице (уколико је прекидач за "*Pomodoro*" укључен, ова опција не постоји, то јест није приказана), која када се прекрије мишем (енгл. *hover*) замењује дати текст са блоком у коме постоји форма, која омогућава кориснику да дефинише дужину трајања временског интервала рада и дужину трајања паузе или једно од та два, и дугме за потврду унетих података (*Слика 2*, *Слика 3*).

Протекло време се може пратити користећи картицу у доњем десном углу са натписом "*PROGRESS"*, која редовно ажурира укупан прогрес корисника за тај дан. Преласком миша преко ње, отвара са књижица на чијој другој страни је приказано протекло време у сатима, минутима и секундама. У сваком тренутку су на челу главне странице екстензије приказани мотивациони цитати који се насумично смењују на сваких тридесет секунди истеклог времена.

За израду књижице са информацијом о дневном напретку корисника, опцију одабира своје технике, дугмиће за контролисање тајмера и још неке делове интерфејса коришћене су *CSS* анимације, понекад у комбинацији са *JavaScript* кодом.

На слици *Слика 13.* приказан је изглед странице када је корисник улогован. Притиском на корисничку иконицу у горњем десном углу пали се искачући прозор у коме постоји опција за брисање историје и дугме за одјављивање из система. Када корисник није улогован, на страници не постоји графикон, јер су графикони специфични за корисника и представљају његов укупан успех у виду статистике на основу броја сати учења по данима, и уместо корисничке иконице стоје дугмад за логовање и регистрацију корисника (*Слика 1*).

При притиску на неко од та два дугмета пали се модал са формом за логовање или регистрацију (*Слика 10*, *Слика 11*). Оба модала садрже форму која захтева потребне податке и дугме за потврду унетих података. Форма за логовање поред тога има и дугме које омогућава кориснику да се улогује на систем преко података са активног *Google* налога.

Због безбедносних ограничења, уграђивање *JavaScript* кода у *HTML* није дозвољено унутар екстензија за *Chrome*, тако да је потребно креирати засебну *JavaScript* датотеку која ће садржати логику те странице. Та датотека мора бити референцирана из *HTML* датотеке под тагом <*script*> на следећи начин:

```
<script src="./content.js"></script>.
```

5.4. Имплементација логике

О позадинској скрипти је било речи у поглављу 4.4.1. Позадинске скрипте се региструју у манифесту под пољем "*background*" као што је приказано у манифесту. Оне раде у позадини, као што и само име говори, што значи да екстензија не мора бити отворена да би позадинске скрипте биле активне. У овом пројекту постоји једна позадинска скрипта ***background.js***. Ова скрипта коришћена је за отварање странице екстензије у новој картици на клик на иконицу екстензије, а то је изведено на следећи начин:

```
chrome.browserAction.onClicked.addListener(function (tab) {
  chrome.tabs.create({ url: "content.html" });
});
```

BrowserAction реферише на иконицу екстензије у адресном бару, а позадинска скрипта прати дешавања са иконицом помоћу слушаоца догађаја. Када се деси догађај прецизиран у коду, у овом случају - кликтање иконице, позива се функција наведена као аргумент функције *addListener* и креира се нова картица користећи chrome.tabs API дозвољен у манифесту, са *URL* адресом главне странице екстензије.

О позадинској скрипти биће речи и мало касније у обради логовања корисника у систем на посебан начин који не захтева прављење налога кроз форму за логовање.

Након што је укључена JavaScript скрипта у *HTML* страницу на приказан начин, функционалности се имплементирају у JavaScript документу дохватањем специфичних компоненти којима су функционалности додељене. Компонентама се у *HTML* коду придружују јединствени идентификациони записи (енгл. *ID*), који омогућавају приступање тим компонентама и дефинисање њиховог рада. Приступ из JS скрипте постиже се на следећи начин:

```
document.getElementById("element-ID").addEventListener("click", () => {
  // ...
});
```

где "*element-ID*" представља поменути идентификациони запис конкретне компоненте, а "*click*" акцију на коју екстензија треба да одреагује. Као што се види у приложеном коду, компоненти се придружује слушалац догађаја (енгл. e*vent listener*), који ослушкује и чека да се деси акција дефинисана првим параметром функције *addEventListener*. Када слушалац догађаја детектује да се дата акција десила, он позива функцију дефинисану другим параметром функције *addEventListener*. Као параметар, функција је углавном у виду *lambda* функције (безимена функција), а њено тело дефинише реакцију на конкретан догађај.

Постоји још један начин да се дохвати компонента из *HTML* кода који је коришћен у овом раду. Елементима *HTML* кода се у оквиру тага, који дефинише тип компоненте, могу доделити разни атрибути, међу којима је између осталог и сам идентификациони запис из претходног пасуса под називом "*id*". Један од атрибута је и атрибут "*class*" који се најчешће употребљава да дефинише класу елемента на основе које ће он бити дизајниран у *CSS* датотеци. Овај атрибут може се користити и као приступна тачка до компоненте из *JavaScript* датотеке. Оно на шта треба припазити у том случају јесте да та класа буде јединствена за специфичну компоненту јер, за разлику од идентификације елемента, исту класу може да дели више елемената, чиме би дошло до конфликта на који се елемент мисли и самим тим до грешке у апликацији. Из тог разлога овај начин је доста ређе коришћен. Он изгледа овако:

```
document.querySelector("element-class").addEventListener("click", () => {
  // ...
});
```

где "*element-class*" представља класу конкретне компоненте.

У наредном тексту биће приказане имплементације одређених функционалности и укратко објашњена идеја која стоји иза тих имплементација.

Као што је већ наведено, притиском на иконицу екстензије отвара се главна страна екстензије у новој картици. Када се отвори главна страница екстензије кориснику се приказује интерфејс објашњен у претходном поглављу. Уколико корисник не жели да се улогује у систем, како би чувао и пратио историју, већ жели тренутно да користи апликацију, омогућене су му скоро све функционалности као и улогованом кориснику.

Ако корисник ипак одлучи да буде улогован и искористи све функционалности, уколико нема свој налог, може га направити притиском на дугме "*Register*" у горњем десном углу поред дугмета за логовање. Кликом на дугме, отвара се модал (енгл. *modal box*) са формом за унос корисничког имена и лозинке. Након што је унео податке, потврђује их кликом на дугме за потврду и уколико су подаци исправни налог бива направљен, а корисник улогован у систем.

За чување података користи се локално складиште *localStorage*, чије коришћење се одобрава дозволом у манифесту. Локално складиште складишти податке у виду стрингова, али захваљујући могућности коришћења *JSON* функције *stringify* и *parse* у складиште се могу смештати и објекти, низови објеката, итд.

Складиште је организовано по принципу хеш (енгл. *hash*) мапе - улазима се приступа помоћу кључева. На основу таквог типа организације, подаци о корисницима су распоређени по различитим ћелијама, а кључ сваке ћелије је *ID* корисника.

Податак о кориснику је објекат који се састоји из његове лозинке, датума када је последњи пут ажуриран и података о времену у виду низа чији елементи су времена учења за различите датуме исказана у јединици секунд. Након што направи налог његови подаци иницијализују се на почетно стање, то јест датум последњег ажурирања на тренутни датум, а низ временских интервала по данима на празан низ. А график бива обојен сивом бојом, која означава да је низ празан, то јест да је корисник укупно учио 0 сати од тренутка када је направио налог.

Ако је корисник већ имао налог, онда уместо на "*Register*" треба да кликне на дугме "*Login*" како би се улоговао. Након клика појављује се модал са формом за логовање, која се састоји из корисничког имена, лозинке, дугмета за потврду унетих података и *Google* дугмета.

Google дугме је дугме које пружа могућност аутентикације корисника преко активног *Google* налога. Уколико корисник први пут искористи ту опцију, подаци ће се иницијализовати на почетно стање и унети у локално складиште. Подаци ће бити сачувани након првог логовања заувек, тако да сваки следећи пут када буде искористио опцију за логовање преко *Google* налога, имаће увид у своје податке. У наредном тексту биће детаљно објашњено како је постигнуто дохватање података са *Google* налога.

OAuth2 је индустријски-стандардни протокол за ауторизацију. Пружа механизам корисницима да дају веб и десктоп апликацијама приступ приватним информацијама без дељења свог корисничког имена, лозинке и других приватних креденцијала. За приступ *Google* налогу корисника коришћен је *Chrome Identity API*. Пошто се екстензије не

учитавају преко *HTTPS*, не могу да врше преусмеравања или постављају колачиће, ослањају се на *Chrome Identity API* да би користили *OAuth2*.

Дозвола за "*identity*" и "*identity.email*" се наводи у манифест фајлу као што је приказано у поглављу 5.2. Поред дозволе потребно је додати још два поља у манифест датотеку. За додавање првог поља потребно је одрадити следећи поступак:

- Потребно је компресовати директоријум екстензије у *.zip* датотеку и отпремити је на *Chrome Developer* таблу (енгл. *dashboard*) без објављивања,
- Затим у *Package* секцији одабрати опцију "*View public key*", из искачућег прозора треба копирати јавни кључ (енгл. *public key*) екстензије и додати га у манифест унутар распакованог директоријума под поље "*key*".

За додавање другог поља потребно је одрадити следећи поступак:

- У *Google API* конзоли креирати нови пројекат; када буде спремно, треба изабрати креденцијале (енгл. *credentials*) у бочном менију, а затим кликнути на "*Create credentials*" и изабрати *OAuth client ID*,
- На страници "*Create client ID*" треба изабрати *Chrome App*, попунити поље за назив екстензије и поставити *ID* екстензије на крај *URL* у пољу *Application ID;* и одабрати опцију за креирање "*Create*", након чега ће конзола обезбедити *OAuth client ID.*
- Укључити поље "*oauth2*" у манифест екстензије; поставити генерисани *OAuth client ID* под део "*client_id*", а под део "*scopes*" додати путању "*https://www.googleapis.com/auth/contacts*". [33]

Након што је претходно извршено, до информација о кориснику долази се на следећи начин:

```
chrome.identity.getProfileUserInfo({ accountStatus: "ANY" }, (userInfo) => {
  chromeUser = userInfo.email;
});
```

где се конкретно дохвата *email* адреса корисника улогованог на *Google* налог.

Уколико корисник којим случајем није улогован на *Google* налог, а затражио је опцију логовања на систем на тај начин, отвара се нови прозор са формом за логовање на *Google* налог. Чим корисник унесе податке и успешно се улогује на *Google* налог, аутоматски се прозор за логовање на *Google* гаси, а фокус враћа на страницу апликације са преузетим информацијама о сада активном *Google* налогу.

За затварање странице за логовање на *Google* налог, након што се корисник успешно улогује, задужена је позадинска скрипта *background.js*, која прати приступање одређеним страницама, на основу тога реагује и затим обавештава скрипту садржаја *content.js* да су потребни подаци унети слањем поруке, које је обрађено у поглављу 4.5.2.:

```
chrome.runtime.sendMessage(message);.
```

А скрипта садржаја када прими поруку отвара налог на систему на основу примљених података и затвара страницу за логовање на *Google*. Примање поруке дато је у наставку.

```
chrome.runtime.onMessage.addListener((msg) => {
  // ...
});
```

У овом случају, кључ за складиште, то јест идентификациони запис корисника је *email*.

Тиме су прећени сви начини на које корисник може да се улогује и да чува своје податке како би могао да контролише своје време захваљујући евиденцији учења по данима.

Мерење времена на тајмеру урађено је коришћењем функције *timerCycle()*. Након што се стартује тајмер притиском на дугме "*Start*" позива се *timerCycle*. Све док се не притисне неко од преостала три дугмета, одбројава се време на тајмеру. Време се зауставља и уколико истекне задати интервал, било да је време рада или пауза у питању. Време се одбројава помоћу функције *setTimeout()* којој се прослеђују два параметра:

- први параметар: функција коју позива,
- други параметар: време изражено у секундама, које представља време за које ће функција задата првим параметром бити позвана.

Помоћу функције *setTimeout*, уколико није на било који начин заустављено време, позива се управо функција *timerCycle* која ажурира време тајмера, како у коду и локалном складишту, тако и на интерфејсу екстензије. Функција *setTimeout* позива се на крају функције *timerCycle* након што обави све остале задатке и пише се на следећи начин:

```
setTimeout("timerCycle()", 1000);.
```

Глобална метода *setTimeout()* поставља тајмер који извршава функцију или одређени део кода када тајмер истекне. То је асинхрона функција, што значи да функција тајмера неће паузирати извршавање других функција у стеку функција. Другим речима,

setTimeout не може да се користи да направи "паузу" пре него што се покрене следећа функција у стеку функција.

Функција *timerCycle* садржи математику потребну за разврставање сати, минута и секунди, које онда уписује у *HTML* компоненту која представља тајмер на екрану. Врши се провера да ли је истекло време паузе или време рада. Ако јесте, тајмер се зауставља, време ресетује и мења режим са паузе на рад или обрнуто. Уколико је истекло време рада, чува се дати интервал времена у историју и припрема систем за започињање паузе. Ако је истекла пауза, време се не чува. Када истекне један од интервала зазвони аларм који се имплементира на следећи начин:

- У директоријум пројекта дода се мелодија аларма са екстензијом *mp3* и затим се у манифесту дефинише у пољу "*web_accessible_resources*",
- У коду се прави аудио објекат и позива када истекне време:

```
var myAudio = new Audio(chrome.runtime.getURL("./sounds/alarm-sound.mp3"));
myAudio.play();.
```

Када истекне интервал за рад, онемогућују се сви дугмићи осим дугмета "*Start*" и чека се корисник да започне паузу.

Графикон је уграђен користећи библиотеку *Chart.js* у објекат са називом *myChart* класе *Chart* креиран на основу *HTML* компоненте која представља тај графикон. Тип овог графикона је пита (енгл. *pie*). [34] За руковање са графиконом задужена је функција *updateChart()* која ажурира *myChart* увек када је то потребно то јест:

- Први пут: на почетку новог дана,
- Након сваког новог завршеног интервала тог дана и
- Последњи пут на крају тог дана, као и у случајевима:
- Када се направи нови корисник или када корисник обрише историју.

Обезбеђено је да се при сваком поновном учитавању странице безбедно поврате подаци помоћу:

```
window.addEventListener("load", () => {
  // ...
});
```

који ослушкује догађај који се дешава са прозором (енгл. *window*), у овом случају догађај учитавања (енгл. *load*).

Такође, корисник у било ком тренутку може да се излогује из система и врати у режим госта (енгл. *guest*).

6. Закључак

У овом раду приказана је израда софтверског додатка веб претраживача *Google Chrome*. За израду су коришћене веб технологије *HTML* (*HyperText Markup Language*), *CSS* (*Cascading Style Sheets*) и *JavaScript*, а као развојно окружење *Visual Studio Code*.

Крајњи резултат рада је алат који служи да помогне студентима да побољшају квалитет учења помоћу понуђених различитих варијанти временске организације. Екстензија омогућава корисницима да поделе своје време на фиксне интервале, раздвојене кратким паузама, и на тај начин постигну максималну ефикасност учења. Поред временске организације, екстензија улогованим корисницима омогућава чување историје учења распоређене по датумима и даје евиденцију у исту у виду графикона.

У раду је приказано креирање манифест датотеке, као и датотека често коришћених у изради екстензије за веб претраживач *content.js* и *background.js*. Приказани су неки од основних концепата на којима почива рад екстензија као што су догађаји, слушаоци догађаја и прослеђивање порука. Дат је увид у неке од могућности екстензије и детаљно објашњена имплементација истих.

За складиштење података у раду је коришћено локално складиште. Надоградња система заснивала би се у том погледу да се подаци чувају динамички уз помоћ заједничке базе коју деле сви корисници. Идеја је да корисници система буду у могућности да се повежу преко налога на екстензији и међусобно прате и подржавају напредак једни других. Како би се ово проширење система реализовало потребно је користити званичну *Google Firebase* базу. *Google Firebase* база података је база података која се налази у облаку (енгл. *cloud-hosted*) и омогућава чување и ажурирање података свих својих клијената у реалном времену. [35]

Учење представља један од захтевнијих задатака са којима се студент свакодневно сусреће. Честа појава код студената је одлагање учења и губитак мотивације. Боља организација времена би омогућила студентима да лакше распореде своје обавезе и на тај начин постигну жељене резултате. Софтверски додатак веб прегледача за ефикасније учење студената је једноставна и приступачна алатка која пружа остварење тог циља.

Списак скраћеница

UCLA	*University of California, Los Angeles*
ARPAnet	*Advanced Research Projects Agency Network*
TCP/IP	*Transmission Control Protocol/ Internet Protocol*
CERN	*Conseil Européen pour la Recherche Nucléaire*
HTML	*HyperText Markup Language*
CSS	*Cascading Style Sheets*
URL	*Uniform Resource Locator*
PDF	*Portable Document Format*
API	*Application Programming Interface*
JSON	*JavaScript Object Notation*
ECMA	*European Carton Makers Association*
DHTML	*Dynamic HyperText Markup Language*
DOM	*Document Object Model*
ID	*Identity document*
HTTPS	*Hypertext Transfer Protocol Secure*

Списак слика

Слика 1. Приказ екстензије	6
Слика 2. Опција дефинисања технике	7
Слика 3. Дефинисање технике	7
Слика 4. *Pomodoro* укључен	7
Слика 5. *Pomodoro* искључен	7
Слика 6. Интервал рада	8
Слика 7. Интервал паузе	8
Слика 8. Прогрес књижица	8
Слика 9. Приказ прогреса	8
Слика 10. *Register* форма	9
Слика 11. *Login* форма	9
Слика 12. *Google* форма	9
Слика 13. Улогован режим	9
Слика 14. Графикон учења	9
Слика 15. Учитавање екстензије	14
Слика 16. Интерфејс екстензије	22

Литература

[1] The Washington Post, "A Flaw In The Design", https://www.washingtonpost.com/sf/business/2015/05/30/net-of-insecurity-part-1/.

[2] V. Cerf and R. Kahn, "A Protocol for Packet Network Intercommunication", https://www.cs.princeton.edu/courses/archive/fall06/cos561/papers/cerf74.pdf.

[3] "Committed to connecting the world" https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx.

[4] T. Berners-Lee, "WorldWideWeb, the first Web client", https://www.w3.org/People/Berners-Lee/WorldWideWeb.html.

[5] W. Stewart, "Web Browser History", https://livinginternet.com/w/wi_browse.htm.

[6] "What is a web browser?", https://www.mozilla.org/en-CA/firefox/browsers/what-is-a-browser/.

[7] "Extensions", https://developer.chrome.com/docs/extensions/.

[8] "Internet Explorer", https://docs.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/platform-apis/aa753620(v=vs.85).

[9] P. Bright, "Edge browser now has extensions in the latest Windows 10 preview", https://arstechnica.com/information-technology/2016/03/edge-browser-now-has-extensions-in-the-latest-windows-10-preview/.

[10] https://www.bleepingcomputer.com/news/google/google-chrome-88-released-rip-flash-player-and-ftp-support/.

[11] "StayFocusd", https://chrome.google.com/webstore/detail/stayfocusd/laankejkbhbdhmipfmgcngdelahlfoji.

[12] "Todoist for Chrome", https://chrome.google.com/webstore/detail/todoist-for-chrome/jldhpllghnbhlbpcmnajkpdmadaolakh.

[13] "Evernote Web Clipper", https://chrome.google.com/webstore/detail/evernote-web-clipper/pioclpoplcdbaefihamjohnefbikjilc?hl=sr.

[14] "Google Dictionary (by Google)", https://chrome.google.com/webstore/detail/google-dictionary-by-goog/mgijmajocgfcbeboacabfgobmjgjcoja.

[15] "Nimbus Screenshot & Screen Video Recorder", https://chrome.google.com/webstore/detail/nimbus-screenshot-screen/bpconcjcammlapcogcnnelfmaeghhagj.

[16] "RescueTime for Chrome and Chrome OS", https://chrome.google.com/webstore/detail/rescuetime-for-chrome-and/bdakmnplckeopfghnlpocafcepegjeap.

[17] "Weava Highlighter - PDF & Web", https://chrome.google.com/webstore/detail/weava-highlighter-pdf-web/cbnaodkpfinfiipjblikofhlhlcickei.

[18] A. Lleras, "Research", University of Illinois, https://psychology.illinois.edu/directory/profile/alleras.

[19] "Market Share Statistics for Internet Technologies", https://netmarketshare.com/.

[20] https://developer.chrome.com/docs/extensions/mv3/getstarted/.

[21] C. Francesco, "The Pomodoro Technique", http://baomee.info/pdf/technique/1.pdf.

[22] "HTML", https://developer.mozilla.org/en-US/docs/Web/HTML.

[23] "CSS", https://developer.mozilla.org/en-US/docs/Web/CSS.

[24] "JavaScript", https://developer.mozilla.org/en-US/docs/Web/JavaScript.

[25] "Background script", https://developer.chrome.com/docs/extensions/mv2/background_pages/.

[26] "Content script", https://developer.chrome.com/docs/extensions/mv3/content_scripts/.

[27] "Chrome Events", https://developer.chrome.com/docs/extensions/reference/events/.

[28] "Message passing", https://developer.chrome.com/docs/extensions/mv3/messaging/.

[29] "Chrome Extension, About", https://developer.chrome.com/docs/extensions/mv3/faq/#capabilities2.

[30] "JSON", https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON.

[31] "Chrome incompatibilities", https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/Chrome_incompatibilities.

[32] "Visual Studio Code", https://code.visualstudio.com/.

[33] "OAuth2: Authenticate users with Google", https://developer.chrome.com/docs/extensions/mv3/tut_oauth/.

[34] "Chart JS", https://www.chartjs.org/.

[35] "Google Firebase", https://firebase.google.com/.