Using Client- Side Substitution Ciphers to Enhance Password Strength

Design Manual

Student: Kevin Davitt
Supervisor: James Egan
C00215181@itcarlow.ie

Table of Contents

Introduction

The following document will explore the design elements of building the Password Mapper application, the application that will implement the elements discussed in the Research Manual and features defined in the Functional Specification. This document will define;

- Technologies to be used in building the application
- Database requirements
- UML diagrams explored in greater depth than the Functional Specification

The application will be a browser plug-in, which interfaces with a back-end web server. As such, the technologies chosen to build this application, as discussed in Section 2 reflect this choice. The primary technologies chosen include:

• JavaScript; client-side browser scripting language,

• MySQL MariaDB; a database management system,

• Apache; Web Server

• PHP; Server-Side code

Finally, in relation to the needs of the application, a database schema will be examined in Section 4 of this document. This schema will include an ER diagram and show an overview of the proposed layout of the database tables.

All the information compiled in this document will allow the Password Mapper application to be built effectively.

Technology Stack

JavaScript

JavaScript is a language used on the client side in browsers that makes web pages interactive. It's cross-platform and object orientated. JavaScript can be used to change the way webpages look and feel. It can also enhance usability (such as preventing the user from registering with a system if the registration password does not comply with the password policy). JavaScript only runs client side, and so cannot be used to enhance the security of the web application the user is using, but it can aid the user in providing correct input.

JavaScript is used in browser plug-ins to interact with the selected page/tab, this means that with the suggested implementation of the Password Mapper application, the use of JavaScript is required.

MariaDB MySQL

MariaDB is a fork of the MySQL relational database management system. It is open-source and is expected to stay free. It is commercially supported, and community developed, the lead developers were some of the original developers of MySQL.

MariaDB has great performance in terms of speed both when it comes to small databases and larger ones. The Password Mapper application will need to keep track of both users and their corresponding map files for each domain, for this reason, a database is required, and MariaDB has been chosen due to the fact it is free and has good performance.

Apache HTTP Server

Apache HTTP Server is a Web Server that is freely available and has an open-source license. It is one of the most popular web servers. A Web Server is a program that uses http/https to serve client to users upon their request. They are used as part of the client server model.

Apache was chosen for the Password Mapper application as the user of the application needs to be able to login and create an account on the site before using the browser plugin, Apache was also chosen because it is open-source and free.

PHP

PHP is used as it is the server-side language that I am most familiar with. PHP will allow granular control of the web application functionality.

Amazon Web Services (AWS)

AWS is cloud technology and is used to run an Elastic Computer 2 (EC2) instance in the cloud which will host the web application. The resource of this instance is scalable and can be increased as demand for the service increases. The Webserver on this instance can also be configured to use HTTPS for security of data in transit with the latest cryptographic algorithms. Certificates signed by both Amazon (used on the AWS load balancers) and LetsEncrypt (used on the Webserver) will be configured to work with the domain mappa.ie, DNS services will be required for this.

Visual Studio IDE

Visual Studio code is the IDE that will be used to develop this application, it is lightweight, fast and integrates easily with source control solutions.

GitLab

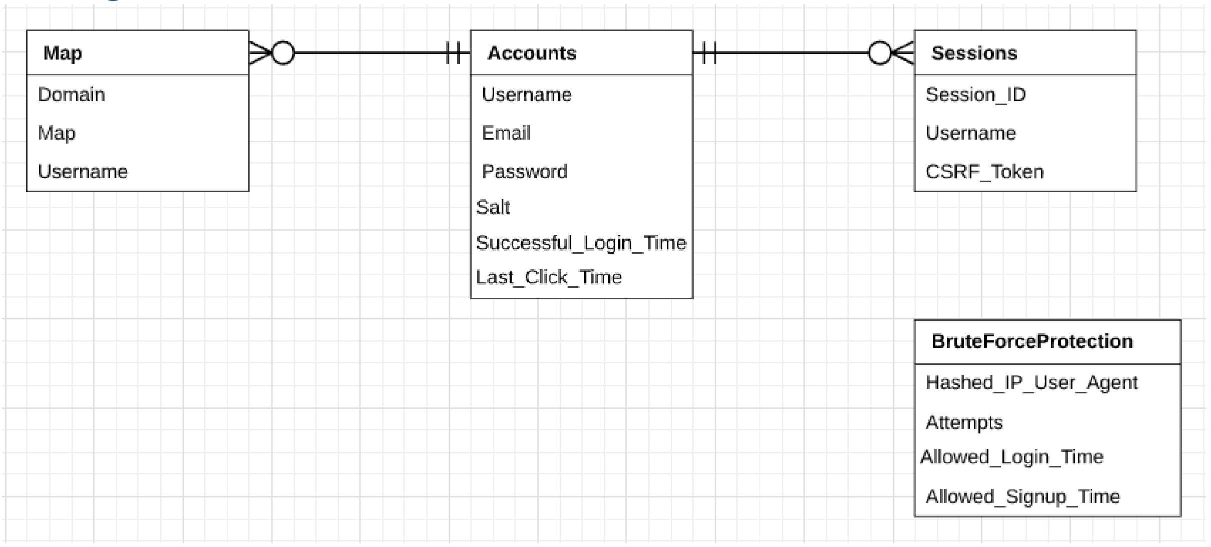
Gitlab is the source control provider used for this project, it enables previous versions of the code to restored easily in the event of issues and enables different machines to be used for development over the course of the project.

Putty SSH Client/WINSCP

Putty and WINSCP are the SSH clients that are used to access the EC2 instance, install the necessary software and set up any configuration files on the server. This is important to get the application up and running but also to harden the Webserver against any attacks.

Database Schema

ER Diagram



Note that these tables may be altered during development, such as the introduction of a logging table. However, this will not affect the relationships shown in the diagram.

Tables

Table Map

Map(VarChar 455)	**Domain (varChar 128)**	**Username (varChar 64)**

Column Usage

Username: Unique Identifier for each user account.
Domain: The domain for which the map is applied.
Map: A string which corresponds to how each key on the keyboard is substituted for other characters.

Table Accounts

Username (varChar 64)	**Email (varChar 96)**	**Account_password (varChar 64)**	**Salt (varChar 128)**

Column Usage

Username: Unique Identifier for each user account.

Email: User’s Email Address.
Password: Password for each user account stored as a hash, salted before hash.
Salt: Used in hash with password to make brute forcing hashes in the case of a breached DB more work

Table Sessions

SessionID (varChar 64)	Username (varChar 64)	CSRF_Token (varChar 64)	Successful_Login_Time (varChar 20)	Last_Click_Time (varChar 20)

Column Usage

Session-ID: Unique identifier for a user’s authenticated session.
Username: Unique Identifier for each user account.
CSRF_Token: Used to protect against Cross Site Request Forgery.
Successful_Login_Time: Will be used for Absolute Timeout.
Last_Click_Time: Used for Idle Timeout.

Table BruteForceProtection

Hashed_IP_User_Agent (varChar 64)	Attempts (smallint)	Allowed_Login_Time (varChar 20)	Allowed_Signup_Time (varChar 20)	Device_csrf (varChar 64)

Column Usage

Hashed_IP_User_Agent: “Unique” identifier for client requests.
Attempts: Keep track of number of authentication attempts, if exceeds X, block requests for Y amount of time.
Allowed_Login_Time: The time from which login requests will be processed again.
Allowed_Signup_Time: The time from which signup (create user) requests will be accepted again.
Device_csrf: An anti-csrf token used to protected unprivileged actions from abuse, such as sign up and sign in functionality.

Table Logging

event_time(varchar 20)	**event_description (var char 32)**	**Outcome(varChar 32)**	**Hashed_IP_User_Agent(varchar 64)**

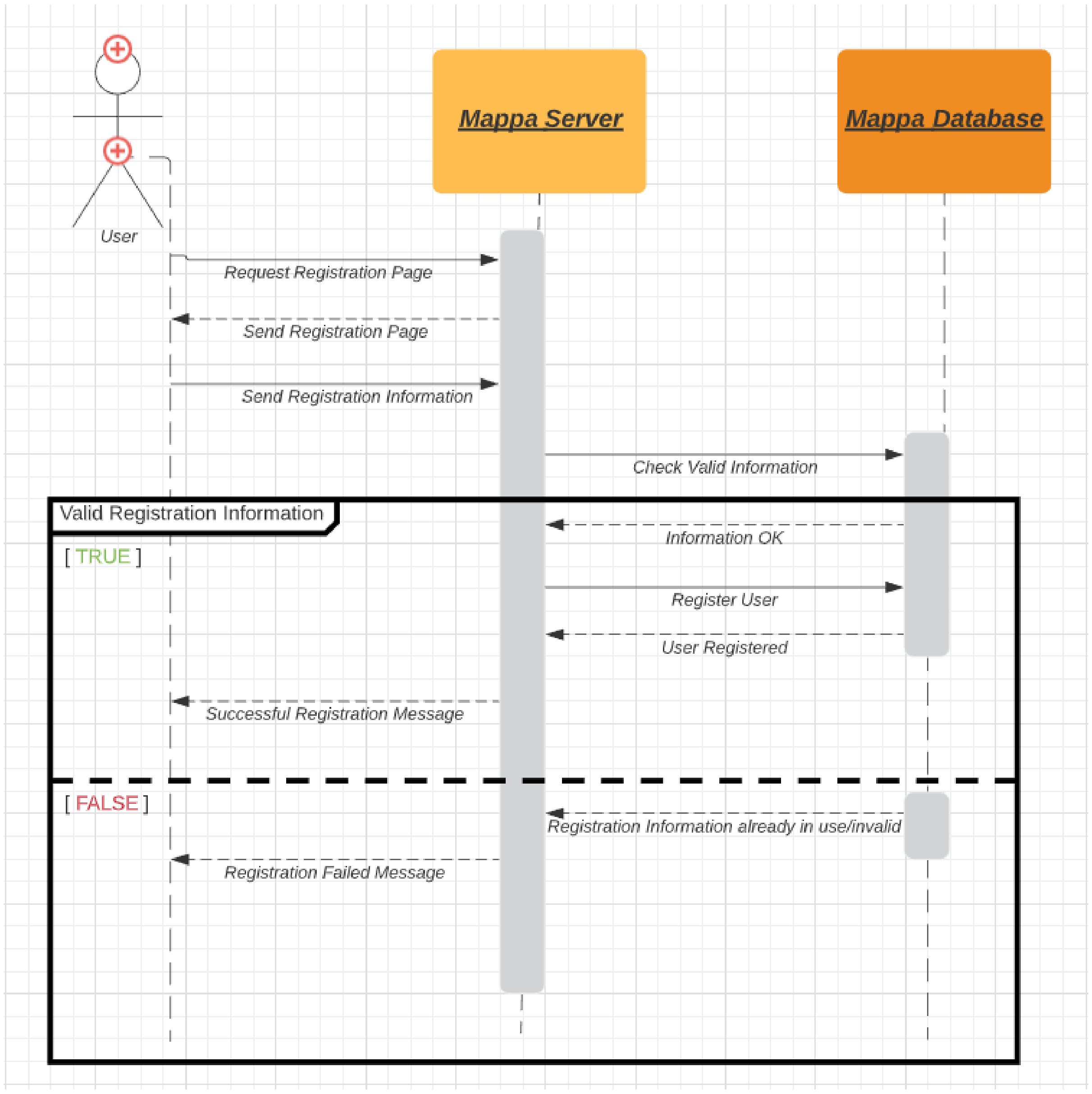
Column Usage

Time: Time of event
Event: Description of what took place
Outcome: Whether attempted action was successful or not
Hashed_IP_User_Agent: Unique identifier for request

Sequence Diagrams

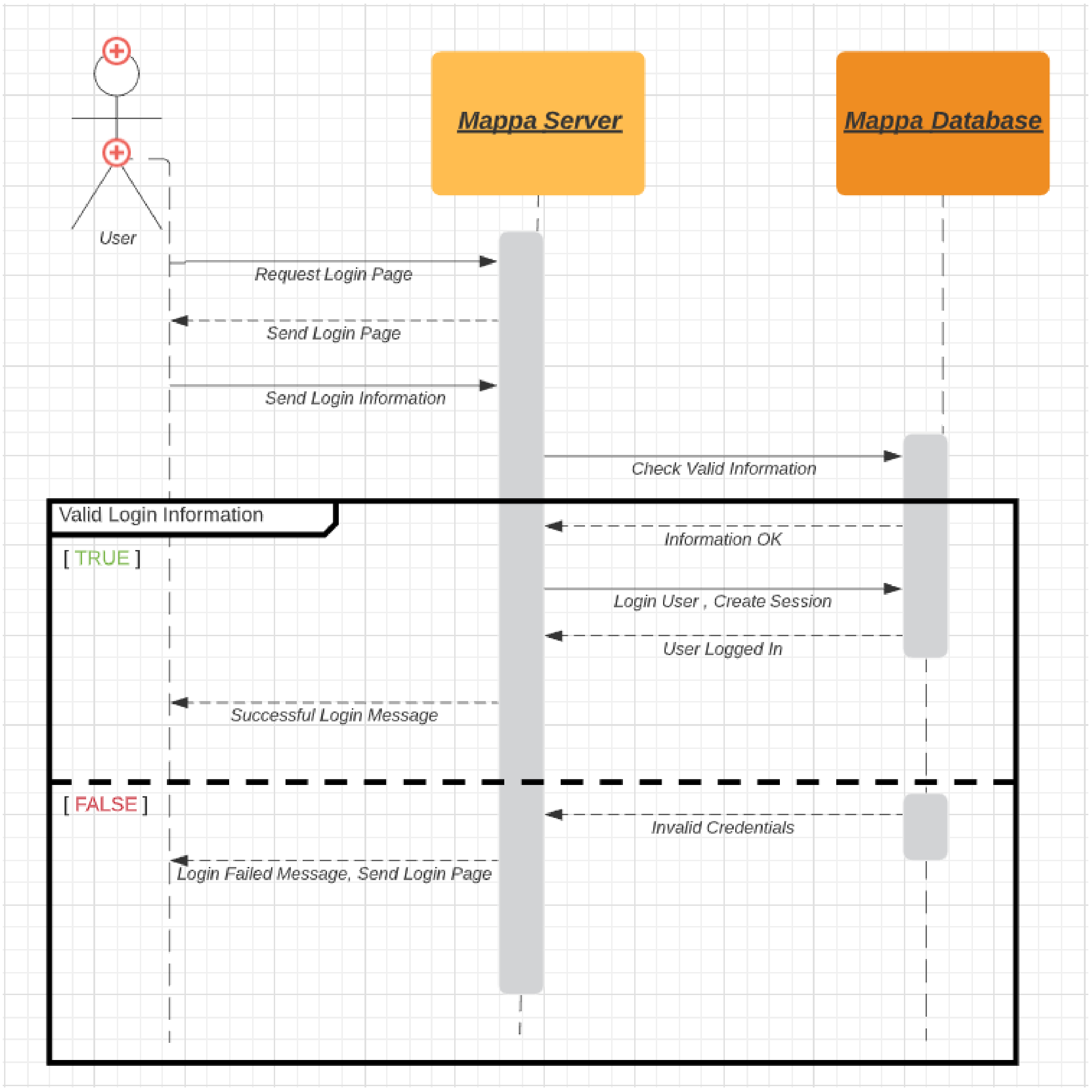
Sign Up

The below sequence diagram shows the process behind signing up for the service.


User
Mappa Server
Mappa Database
Request Registration Page
Send Registration Page
Send Registration Information
Check Valid Information
Valid Registration Information
[TRUE]
Information OK
Register User
User Registered
Successful Registration Message
[FALSE]
Registration Information already in use/invalid
Registration Failed Message

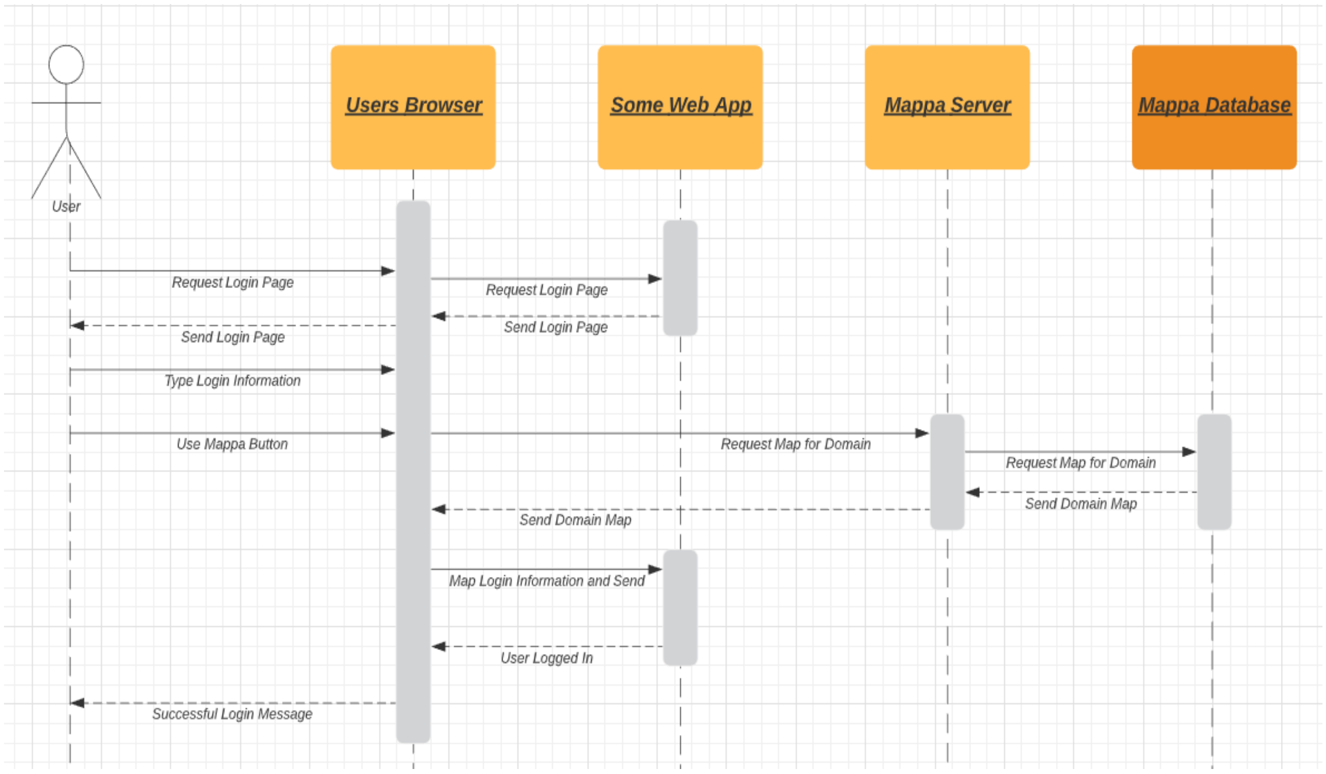
Login

The below sequence diagram shows the process behind authenticating with the application.

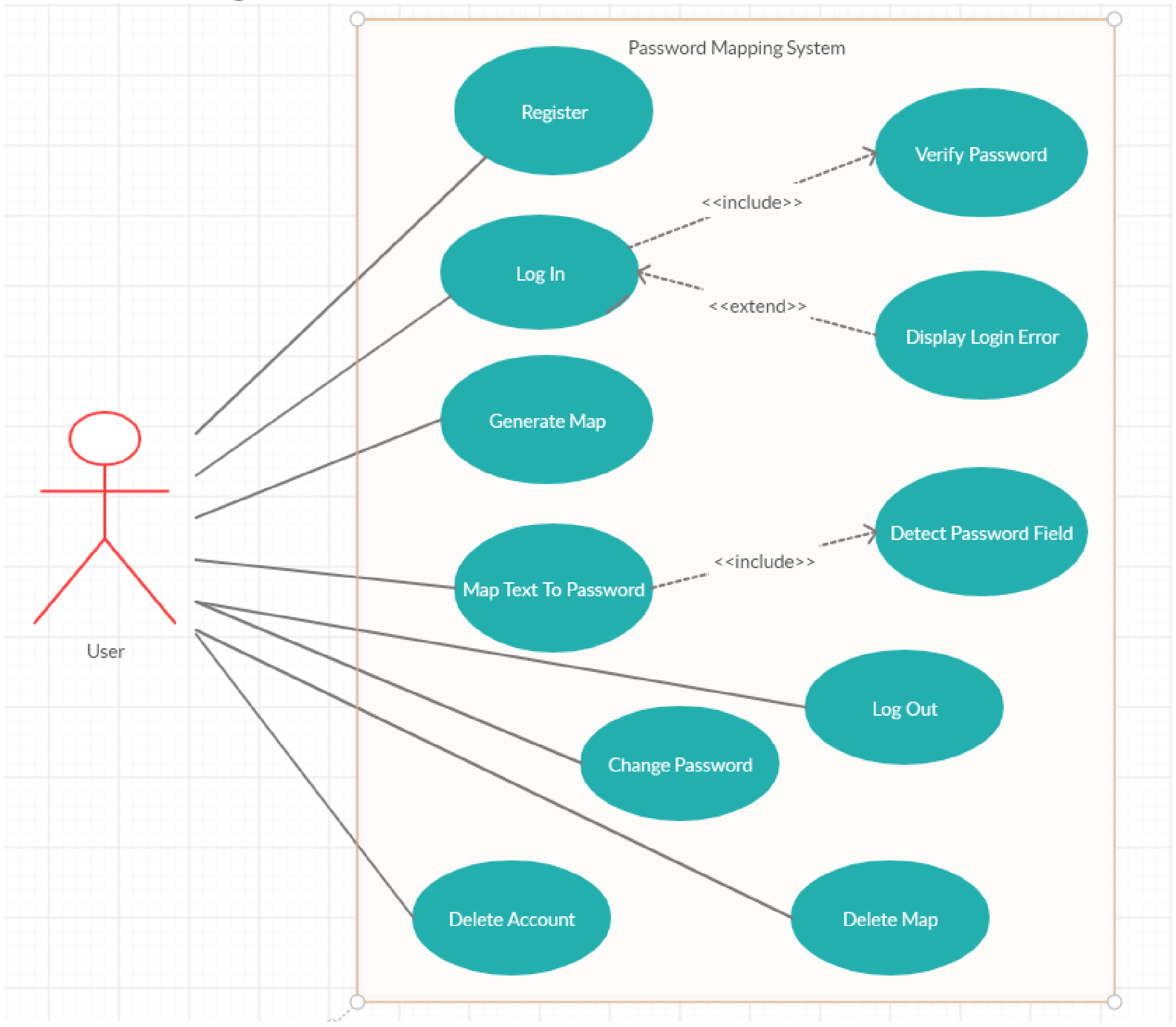


General Use

The process below illustrates the general use case of Mappa, an entirely similar process is used to integrate 3rd party services with Mappa and a more explicit walkthrough is available in the User Manual. It has been omitted from this document due to its similarity to the below.



Use Case Diagram

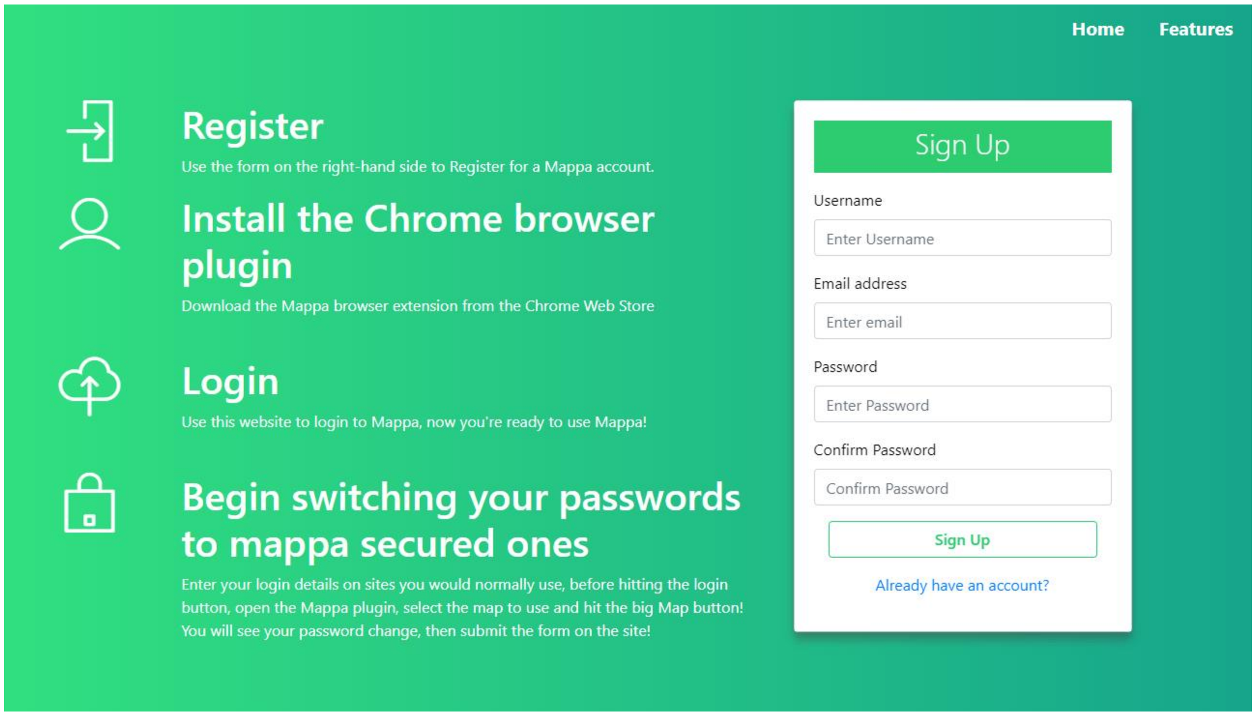


Process Flow – With Interface Screenshots

Webserver Interface

Register

The following flow of screenshots illustrates the functionality of the sign up page for the application.



Register
Use the form on the right-hand side to Register for a Mappa account.

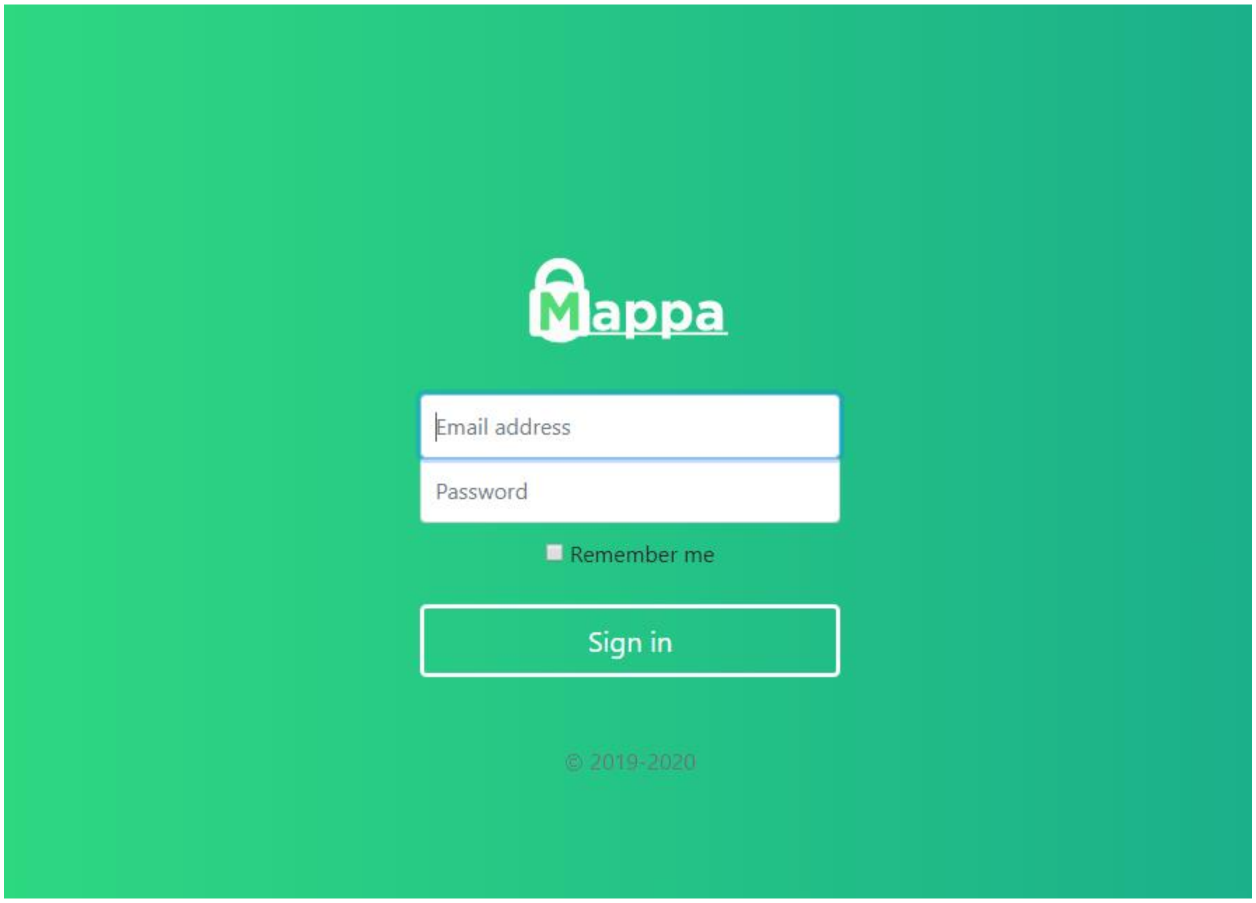
Install the Chrome browser plugin
Download the Mappa browser extension from the Chrome Web Store

Login
Use this website to login to Mappa, now you're ready to use Mappa!

Begin switching your passwords to mappa secured ones
Enter your login details on sites you would normally use, before hitting the login button, open the Mappa plugin, select the map to use and hit the big Map button! You will see your password change, then submit the form on the site!

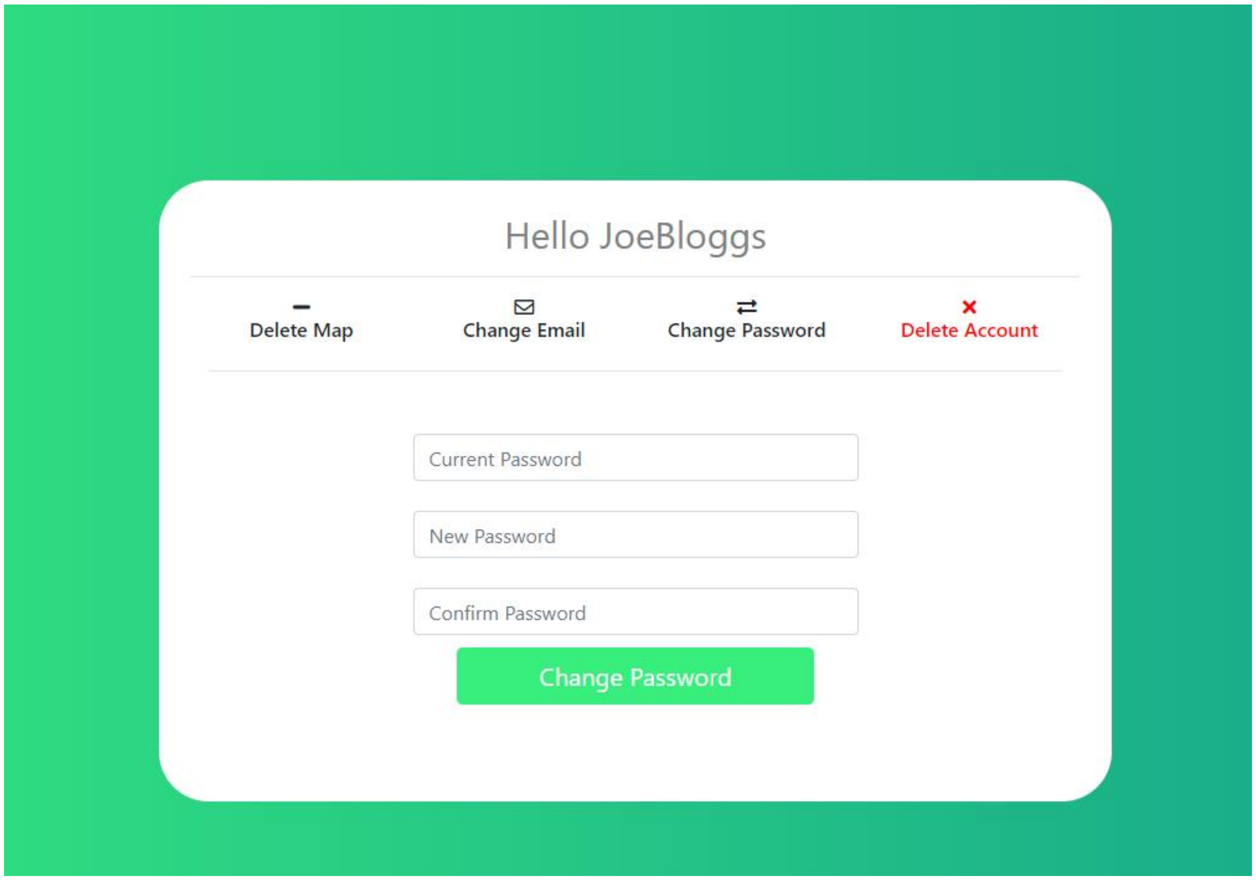
Login

A design for web server login is shown below.



Account Options

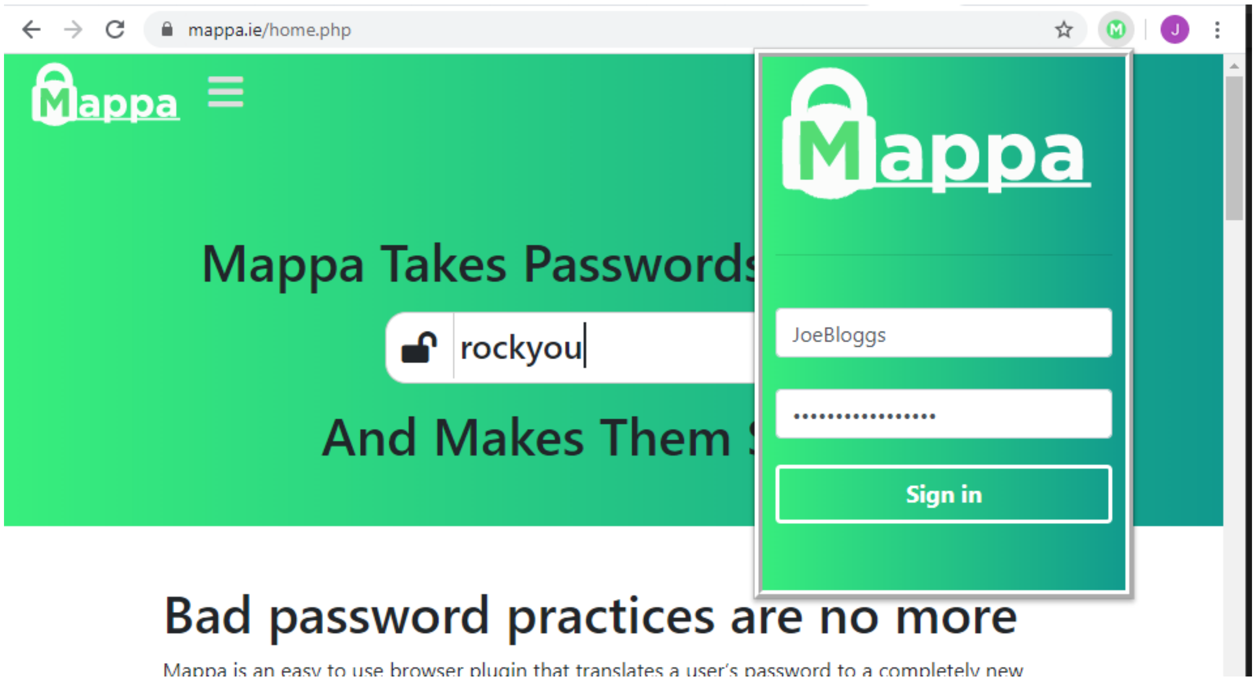
An interface must be available for the user to perform common account functions, such as changing their password. A sample interface design is below.



Browser Extension Interface

Login

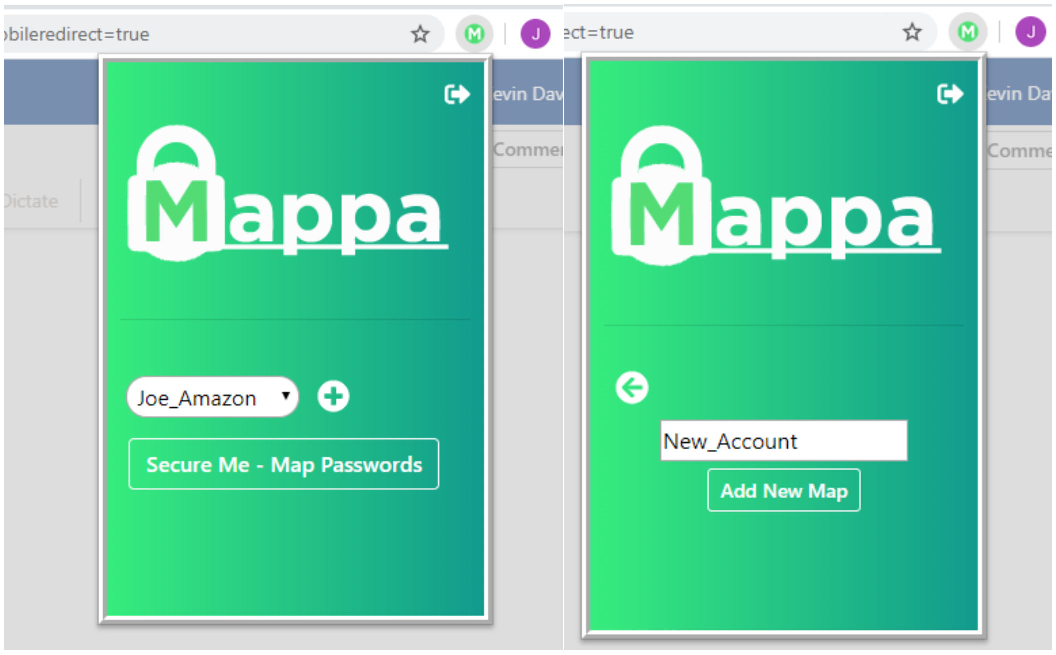
Once you a user has registered for an account, they can log in using the extension. They must click the icon at the top right of their browser and provide their credentials.


mappa.ie/home.php
Mappa
Mappa Takes Passwords
rockyou
And Makes Them
Bad password practices are no more
Mappa is an easy to use browser plugin that translates a user's password to a completely new
Mappa
JoeBloggs
Sign in

Generate Map

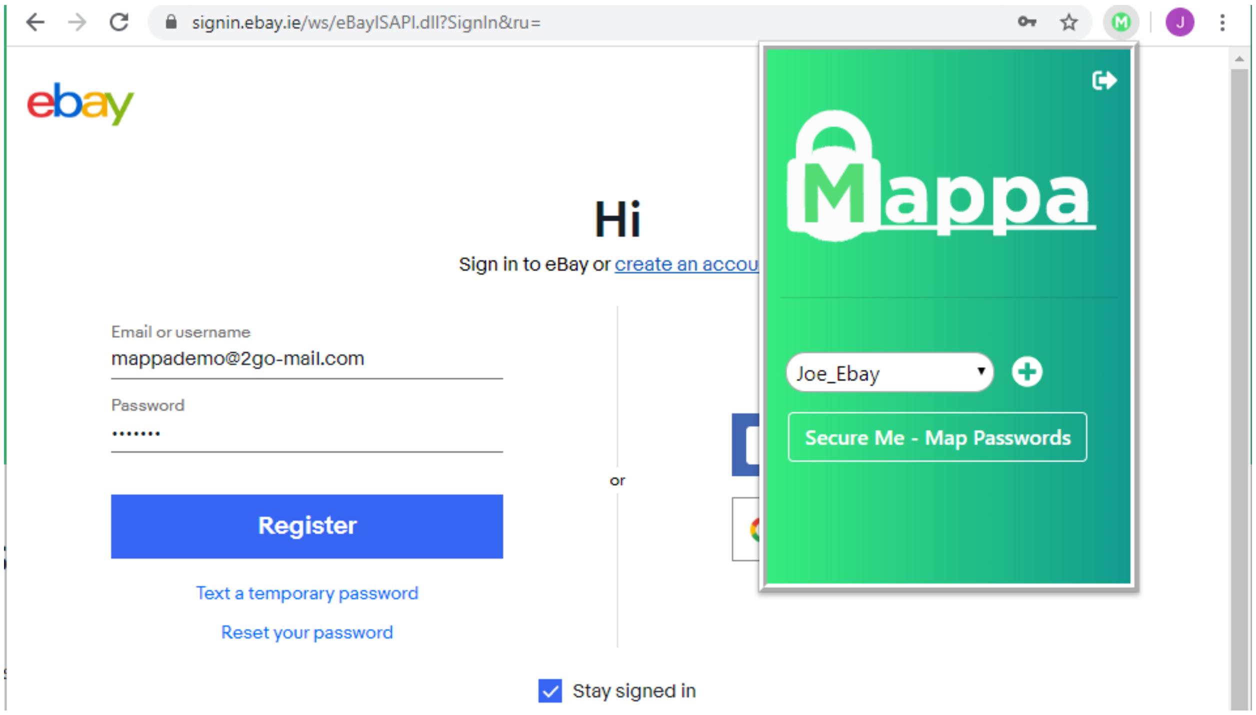
To create a Map, the user must select the + icon, type the desired map name and press the add map button.

A map should be created for every service that a user plans to use Mappa with.

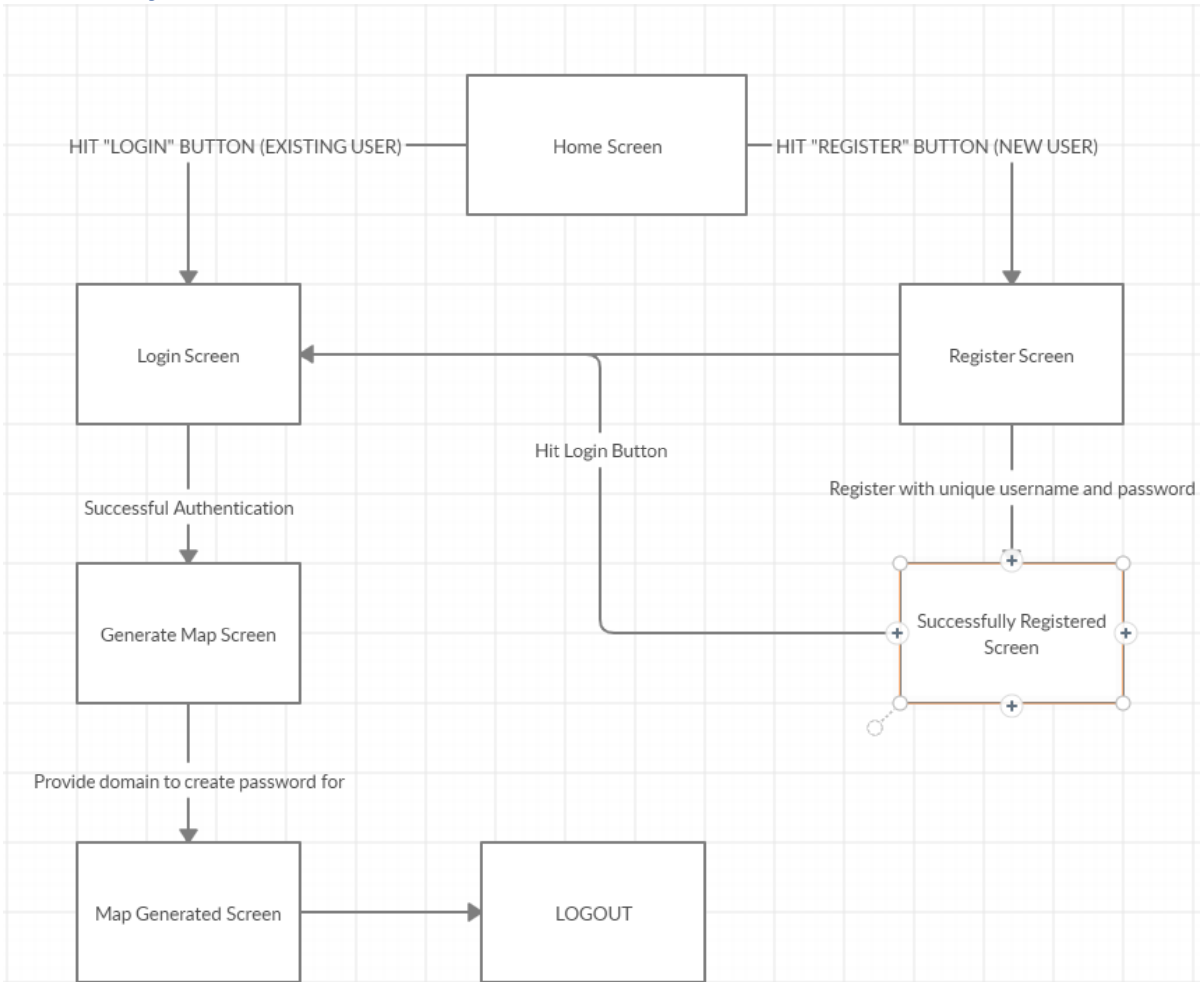


Map Password

To actively use Mappa with services, the user must navigate to the 3rd party site, enter their login information, press the browser extension icon, select the map for the site, and press the Map Passwords button. This process has already been explained in the General Use sequence diagram.


signin.ebay.ie/ws/eBayISAPI.dll?SignIn&ru=
ebay
Hi
Sign in to eBay or create an accou
Email or username
mappademo@2go-mail.com
Password
•••••••
Register
Text a temporary password
Reset your password
or
Stay signed in
Mappa
Joe_Ebay
Secure Me - Map Passwords

User Navigation Flow

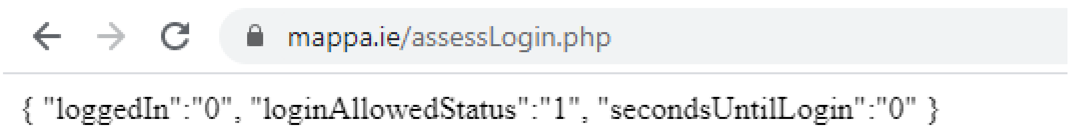

Home Screen
HIT "LOGIN" BUTTON (EXISTING USER)
HIT "REGISTER" BUTTON (NEW USER)
Login Screen
Register Screen
Hit Login Button
Register with unique username and password
Successful Authentication
Generate Map Screen
Successfully Registered Screen
Provide domain to create password for
Map Generated Screen
LOGOUT

Background Functionality

All the following functionality will be implementing using AJAX calls to the application's web server.

Assess Login Status

The extension requires a means to determine whether or not a user is currently logged in, this will done using an ajax call to the web server. The server will respond with a JSON object that the extension can then use to determine what information and functionality to present to the user. An example of a response to this request is given below.



mappa.ie/assessLogin.php

```
{ "loggedIn":"0", "loginAllowedStatus":"1", "secondsUntilLogin":"0" }
```

Generating Maps

For map generation, a similar method is used. The extension will send an ajax request to the server that includes a map name, the server will then generate a map for the user and assign the requested name to it. Details about this function can be seen in the pseudocode in the next section, or in the Technical Manual.

This function allows maps to be created and saved on the server side.

Retrieve Map Names

Once logged in, the extension will send a request to the server to retrieve the map names for the current user. The server will respond with the name of each map and this information will be used by the extension to populate the list box, which allows users to select the map they want to use.

This should all happen automatically and should not be something the user must worry about.

Map Injection

Once a user has pressed the Map Password button, the extension must inject both the Map itself and the mapping script into the active tab, for this, active tab permission is required in the extension manifest file. Additionally, the Map will be saved temporarily to the browser's local storage, in order for a script in the active tab to read it.

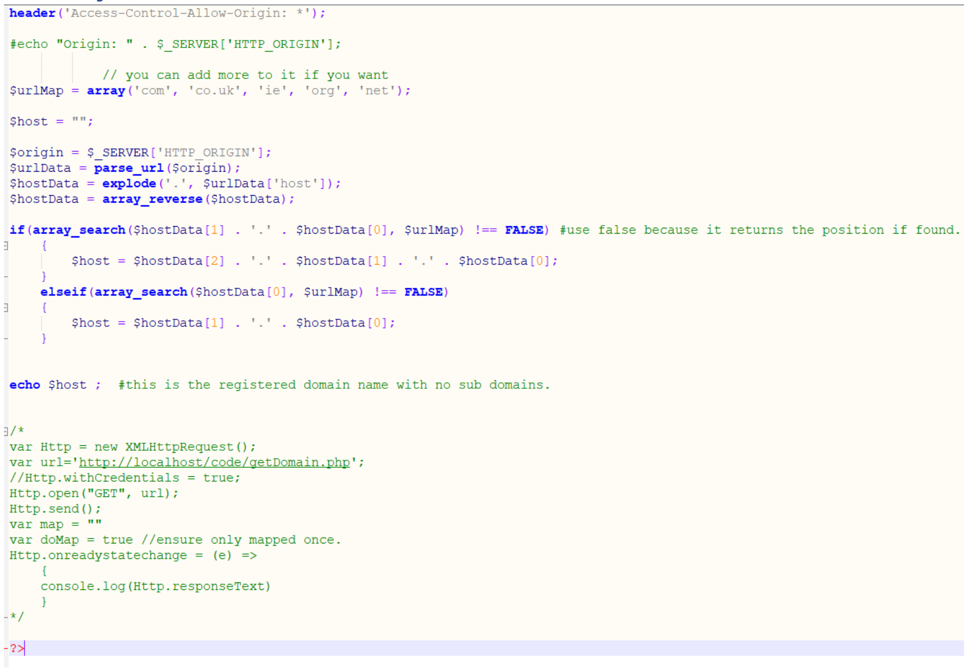
Logging

The system will log privileged actions by saving event information to the logging table described in this document. This table has a simple way to categorize events. A simple function will written that can be used every time an event is logged. More information on this function can be obtained by reading the security functions file in the technical manual.

Pseudocode

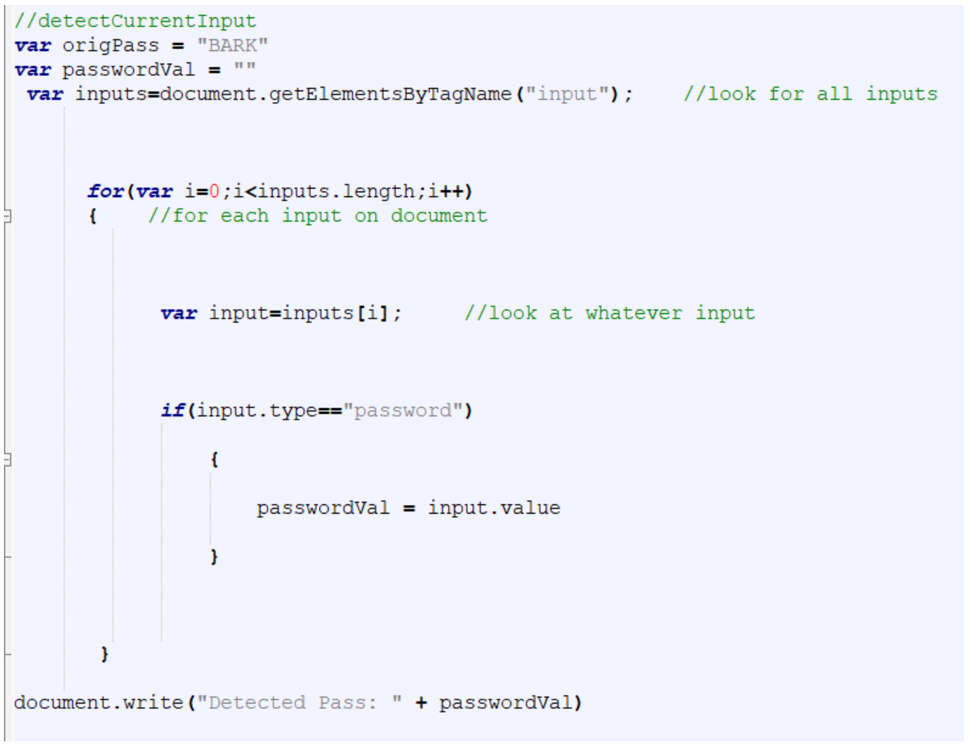
The following section shows code snippets written in a couple of hours that show roughly how certain functions will work. Not all the code here may be used or implemented the exact same way. It simply serves to illustrate the logical process followed.

Identify Domain



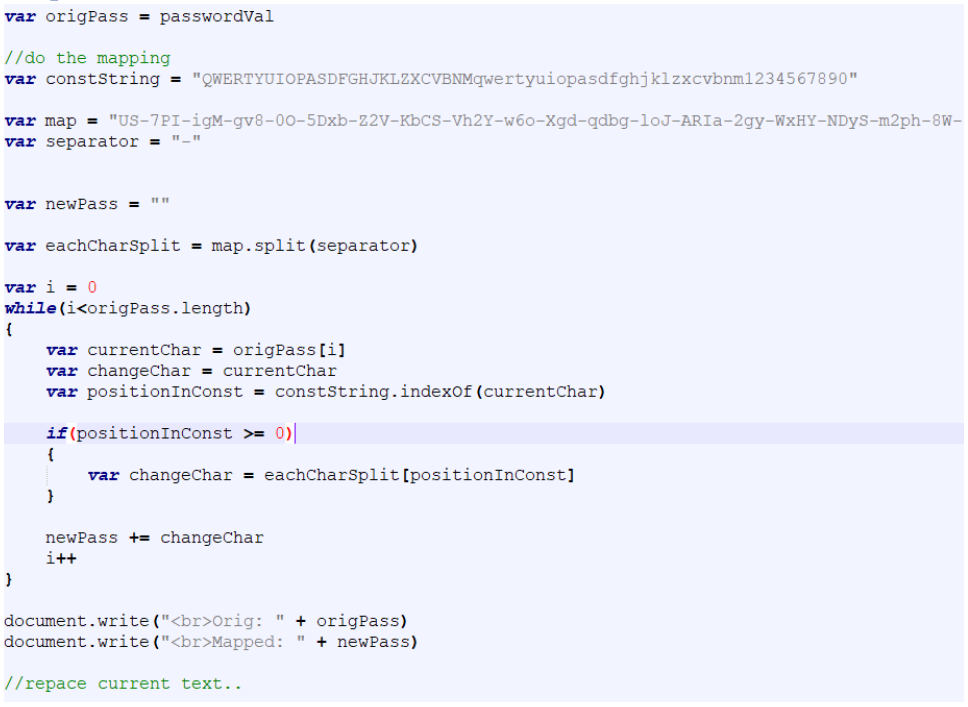
```
header('Access-Control-Allow-Origin: *');

#echo "Origin: " . $_SERVER['HTTP_ORIGIN'];

            // you can add more to it if you want
$urlMap = array('com', 'co.uk', 'ie', 'org', 'net');

$host = "";

$origin = $_SERVER['HTTP_ORIGIN'];
$urlData = parse_url($origin);
$hostData = explode('.', $urlData['host']);
$hostData = array_reverse($hostData);

if(array_search($hostData[1] . '.' . $hostData[0], $urlMap) !== FALSE) #use false because it returns the position if found.
    {
        $host = $hostData[2] . '.' . $hostData[1] . '.' . $hostData[0];
    }
    elseif(array_search($hostData[0], $urlMap) !== FALSE)
    {
        $host = $hostData[1] . '.' . $hostData[0];
    }


echo $host ;  #this is the registered domain name with no sub domains.


/*
var Http = new XMLHttpRequest();
var url='http://localhost/code/getDomain.php';
//Http.withCredentials = true;
Http.open("GET", url);
Http.send();
var map = ""
var doMap = true //ensure only mapped once.
Http.onreadystatechange = (e) =>
    {
    console.log(Http.responseText)
    }
*/

?>
```

Detect Current Text



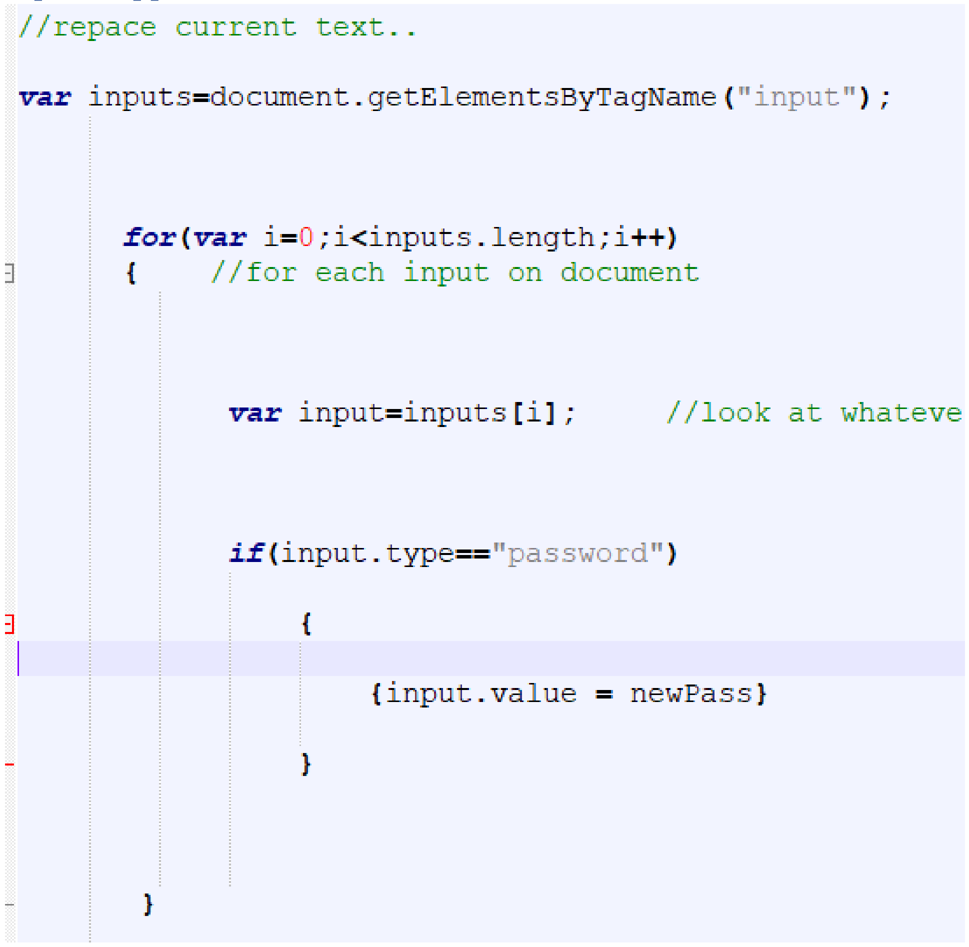
```
//detectCurrentInput
var origPass = "BARK"
var passwordVal = ""
 var inputs=document.getElementsByTagName("input");    //look for all inputs


      for(var i=0;i<inputs.length;i++)
      {    //for each input on document


            var input=inputs[i];     //look at whatever input


            if(input.type=="password")
                {
                    passwordVal = input.value
                }


       }
document.write("Detected Pass: " + passwordVal)
```

Map Text



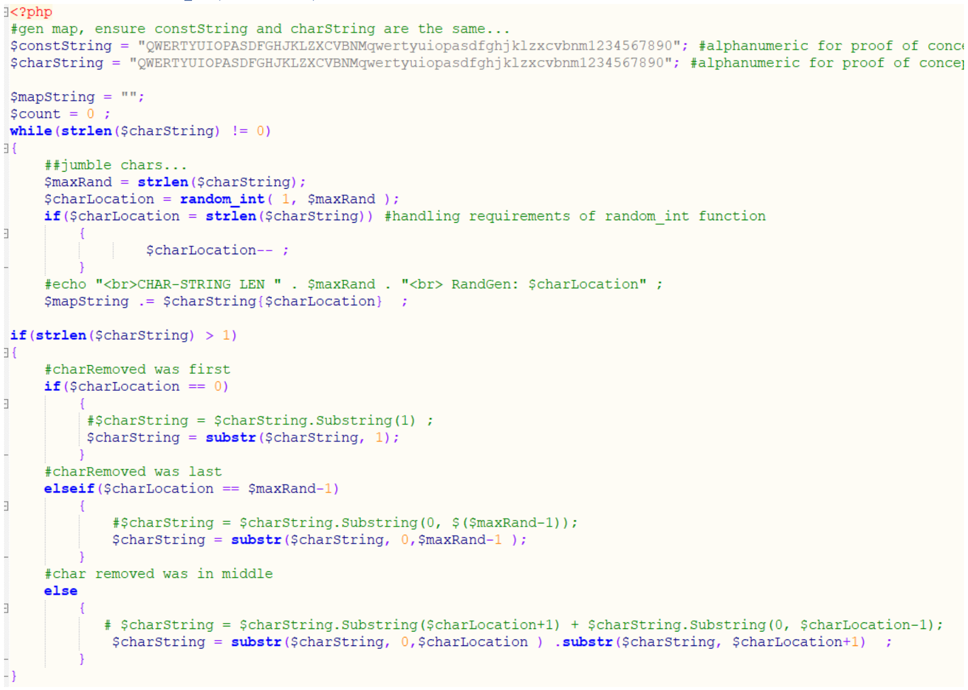
```
var origPass = passwordVal

//do the mapping
var constString = "QWERTYUIOPASDFGHJKLZXCVBNMqwertyuiopasdfghjklzxcvbnm1234567890"

var map = "US-7PI-igM-gv8-0O-5Dxb-Z2V-KbCS-Vh2Y-w6o-Xgd-qdbg-loJ-ARIa-2gy-WxHY-NDyS-m2ph-8W-
var separator = "-"


var newPass = ""

var eachCharSplit = map.split(separator)

var i = 0
while(i<origPass.length)
{
    var currentChar = origPass[i]
    var changeChar = currentChar
    var positionInConst = constString.indexOf(currentChar)

    if(positionInConst >= 0)
    {
        var changeChar = eachCharSplit[positionInConst]
    }

    newPass += changeChar
    i++
}

document.write("<br>Orig: " + origPass)
document.write("<br>Mapped: " + newPass)

//repace current text..
```

Input Mapped Text

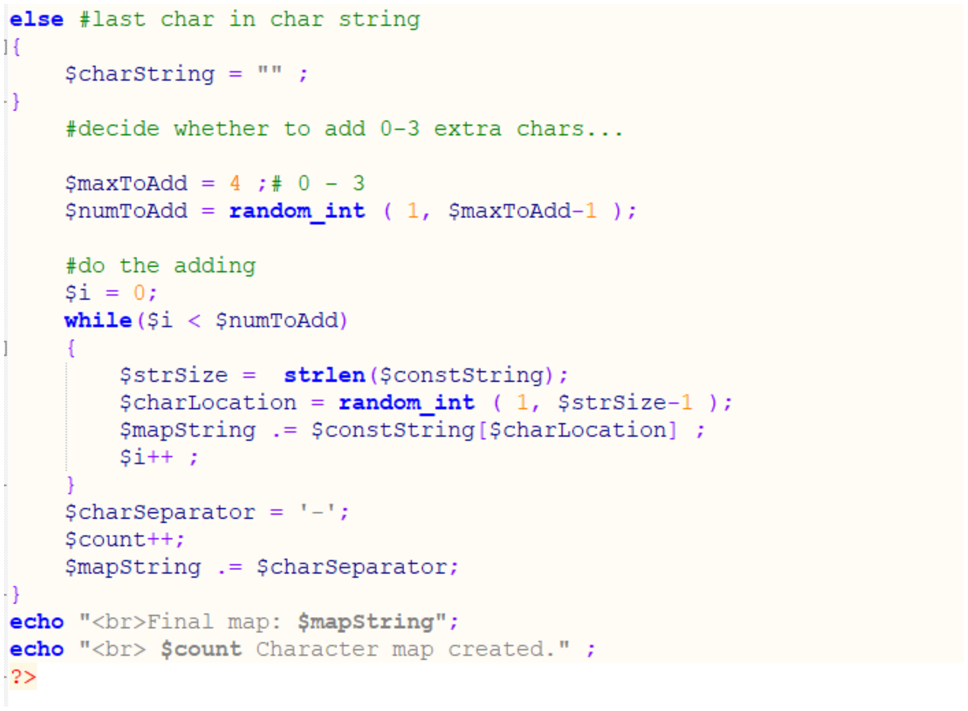


```
//repace current text..

var inputs=document.getElementsByTagName("input");


        for(var i=0;i<inputs.length;i++)
        {     //for each input on document


               var input=inputs[i];      //look at whateve


               if(input.type=="password")

                    {

                         {input.value = newPass}

                    }



          }
```

Generate Map (Server)



```
<?php
#gen map, ensure constString and charString are the same...
$constString = "QWERTYUIOPASDFGHJKLZXCVBNMqwertyuiopasdfghjklzxcvbnm1234567890"; #alphanumeric for proof of conc
$charString = "QWERTYUIOPASDFGHJKLZXCVBNMqwertyuiopasdfghjklzxcvbnm1234567890"; #alphanumeric for proof of conce

$mapString = "";
$count = 0 ;
while(strlen($charString) != 0)
{
    ##jumble chars...
    $maxRand = strlen($charString);
    $charLocation = random_int( 1, $maxRand );
    if($charLocation = strlen($charString)) #handling requirements of random_int function
        {
                $charLocation-- ;
        }
    #echo "<br>CHAR-STRING LEN " . $maxRand . "<br> RandGen: $charLocation" ;
    $mapString .= $charString{$charLocation}  ;

if(strlen($charString) > 1)
{
    #charRemoved was first
    if($charLocation == 0)
        {
         #$charString = $charString.Substring(1) ;
         $charString = substr($charString, 1);
        }
    #charRemoved was last
    elseif($charLocation == $maxRand-1)
        {
            #$charString = $charString.Substring(0, $($maxRand-1));
            $charString = substr($charString, 0,$maxRand-1 );
        }
    #char removed was in middle
    else
        {
           # $charString = $charString.Substring($charLocation+1) + $charString.Substring(0, $charLocation-1);
            $charString = substr($charString, 0,$charLocation ) .substr($charString, $charLocation+1)  ;
        }
}
```



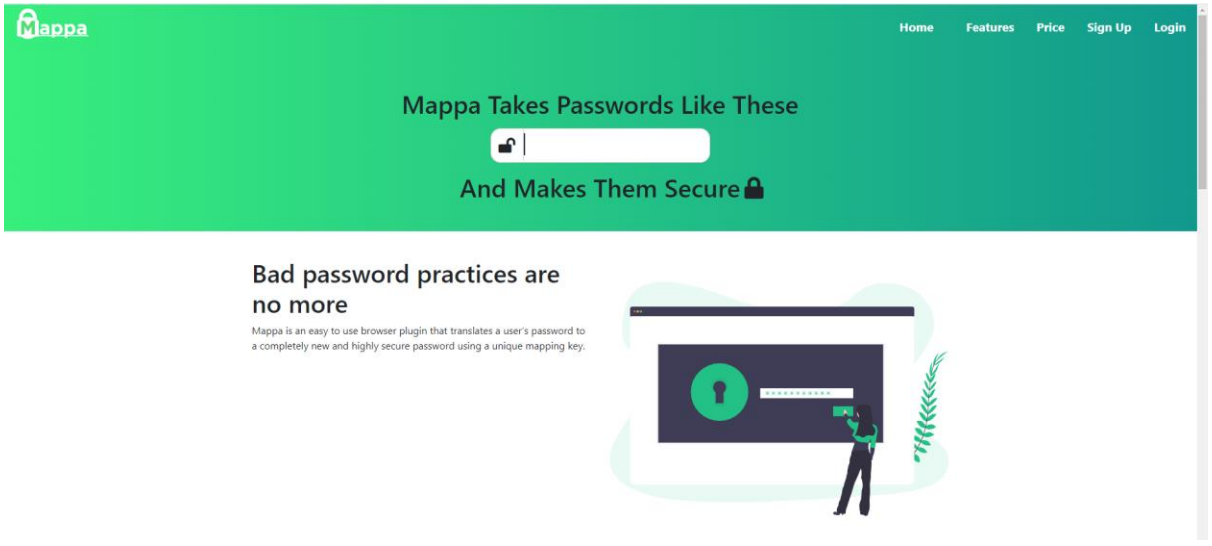
```
else #last char in char string
{
    $charString = "" ;
}
    #decide whether to add 0-3 extra chars...

    $maxToAdd = 4 ;# 0 - 3
    $numToAdd = random_int ( 1, $maxToAdd-1 );

    #do the adding
    $i = 0;
    while($i < $numToAdd)
    {
        $strSize =  strlen($constString);
        $charLocation = random_int ( 1, $strSize-1 );
        $mapString .= $constString[$charLocation] ;
        $i++ ;
    }
    $charSeparator = '-';
    $count++;
    $mapString .= $charSeparator;
}
echo "<br>Final map: $mapString";
echo "<br> $count Character map created." ;
?>
```

Methodology

This project will be approached with a Scrum methodology, producing working software at the end of most sprints. Each function or code files finished by the end of each sprint will be tested at the beginning of the next.

UI/UX and Rationale

Extensive effort and importance will be placed on designing the user interface to be straightforward and easy to follow. Due to this application being targeted at everyday users of web applications, it should be as simple and straight forward to use, with minimal knowledge required and minimal buttons. Bright colours will be used to give the user a sense of easiness and friendliness. Screenshots of the proposed design can be seen in Process Flow section of this document.A sample colour scheme and wireframe for the home page is shown below.



Mappa Takes Passwords Like These

And Makes Them Secure

Bad password practices are no more

Mappa is an easy to use browser plugin that translates a user's password to a completely new and highly secure password using a unique mapping key.

Project Plan

This plan has already been specified in the Function Specification but has been elaborated on here.

Sprint #	Plan	Due Date	Deliverable
0	Complete Necessary Research	31/10/2019	Technical Research Show with Maths whether software can theoretically enhance security
1	Basic Browser Extension work	14/11/2019	Produce an extension that injects a script into the active tab
2	Map Generation Function	28/11/2019	Produce a function that generates cryptographically random character maps
3	Map retrieval and character mapping function	12/12/2019	Produce secure map retrieval and character mapping functions
BREAK	Two Weeks Holiday	26/12/2019	-
4	Integrate previous elements	09/01/2020	Show domain detection, map retrieval and character mapping taking place on arbitrary domains for Update Presentation
N/A	**Presentation to Supervisor**	16/01/2020	Major Milestone, at this stage the application should be ready for a live demonstration, extra time will be taken leading up to this date to touch up anything necessary for the demo
5	Web Interface Back-End	23/01/2020	Functional and secure login and registration page for users

6	Web Interface Front-End	06/02/2020	A front end for the website that is very, very pretty
7	Map Redundancy	20/02/2020	A local application that can download and store a user’s maps as backup in case Mappa goes down
8	User Testing and Vulnerability testing	27/02/2020	Test full application, report of any flaws. Ensure both user and security testing is completed.
9	Debugging and Security	12/03/2020	Fix any major flaws that affect the function of the application, keeping security a top priority
10	Reserved for agility	03/04/2020	Complete any final tasks that have not been completed up to this point, review Final Project. Repeat security and user testing.
11	**Final Submission**	23/04/2020	The final project should be reviewed and any final polishing of the interfaces done. Every opportunity should be taken to obtain feedback from average users.