

## **TEAM NETWORK**

### **TEAM MEMBERS :**

FARHAN P MOHAMMED

KEERTHI T.M

ALEN MORRES

### **PROBLEM STATEMENTS :**

- 1) Detection and prevention of IDN Homograph Attacks (15th November - April 18th)
- 2) UA readiness of IDN Email Address in websites (April 20th - May 15th)

**PROJECT TIMELINE :** Detection and prevention of IDN Homograph Attacks (15th November - April 18th)

# Browser Extension

Printed from Asana

## IDEATION STAGE

- ☒ **Farhan P Mohammed: ~~LEARNING SETUP~~** due Nov 15, 2021  
Priority: Low
- ☒ **Alen Morres: ~~DEVELOPMENT ENVIRONMENT SETUP~~** due Nov 20, 2021  
Priority: Low
- ☒ **Keerthi T M: ~~HOSTING BROWSER EXTENSIONS~~** due Nov 25, 2021  
Priority: Low

## ASSET AND REQUIREMENT DEVELOPMENT

- ☒ **Alen Morres: ~~BRAINSTORMING / MOOD BOARDING~~** due Nov 15, 2021  
Priority: Low
- ☒ **Alen Morres: ~~CREATIVE BRIEF~~** due Nov 15, 2021  
Priority: Low
- ☒ **Alen Morres: ~~SKETCHING , DESIGN BUILDING~~** due Nov 20, 2021  
Priority: Medium
- ☒ **Alen Morres: ~~IDN LOGO AND ASSET DESIGNING~~** due Nov 20, 2021  
Priority: Medium
- ☒ **Alen Morres: ~~UI/UX BRAINSTORMING~~** due Nov 30, 2021  
Priority: Low
- ☒ **Alen Morres: ~~UX/UI DESIGN FLOW , RESPONSIBLE'S AND DELIVERABLES~~** due Nov 30, 2021  
Priority: Low
- ☒ **Alen Morres: ~~REFINING AND REVISIONS~~** due Nov 30, 2021  
Priority: Low
- ☒ **Alen Morres: ~~REPORT GENERATION~~** due Feb 15, 2022  
Priority: High

## INTERIM FEATURES LIST

- ☒ **Keerthi T M: ~~ON/OFF MAIN BUTTON~~** due Nov 30, 2021  
Priority: Medium
- ☒ **Keerthi T M: ~~U-LABEL DISPLAY~~** due Dec 15, 2021  
Priority: Medium
- ☒ **Keerthi T M: ~~A-LABEL DISPLAY~~** due Dec 15, 2021  
Priority: Medium
- ☒ **Keerthi T M: ~~STATUS SELECTION~~** due Dec 15, 2021  
Priority: Medium
- ☒ **Keerthi T M: ~~BLACKLIST SELECTION~~** due Dec 15, 2021  
Priority: Medium

**STATUS SECTION**

- ☒ **Alen Morres: DOMAIN STATUS** due Dec 28, 2021  
Priority: Medium
- ☒ **Alen Morres: SAFE / RISKY / INSECURE WITH SPECIFIC ALERTS** due Dec 28, 2021  
Priority: Medium
- ☒ **Alen Morres: INFO ALERT SECTION** due Dec 28, 2021  
Priority: Medium

**BLACKLIST SECTION**

- ☒ **Farhan P Mohammed: ADD DOMAIN BUTTON** due Jan 5, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: REMOVE DOMAIN BUTTON** due Jan 5, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: WHITELIST BUTTON** due Jan 5, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: SHOW LIST TABLE** due Jan 5, 2022  
Priority: Medium

**IDN DETECC**

- ☒ **Alen Morres: BRAINSTORMING AND WHAT, WHY ANALYSIS** due Jan 10, 2022  
Priority: Low
- ☒ **Alen Morres: PROTOTYPING** due Jan 15, 2022  
Priority: Medium
- ☒ **Keerthi T M: FUNCTIONALITY TESTING, MODULE TESTING AND REFINEMENT** due Jan 30, 2022  
Priority: High
- ☒ **Keerthi T M: FEATURE DEVELOPMENT** due Feb 5, 2022  
Priority: High
- ☒ **Keerthi T M: PRODUCT DEVELOPMENT AND DEPLOY** due Feb 15, 2022  
Priority: High
- ☒ **Farhan P Mohammed: TESTING AND REPORT GENERATION** due Feb 15, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: REPORT AND DOCUMENT REVIEW** due Feb 17, 2022  
Priority: Medium

**IDN PROTECC**

- ☒ **Alen Morres: DESIGN AND FABRICATION OF THE INTERFACE** due Feb 25, 2022  
Priority: High
- ☒ **Alen Morres: PROTOTYPING MECHANICS** due Mar 10, 2022  
Priority: High

- ☒ **Alen Morres: ~~UNIT AND FUNCTIONAL TESTING~~ → REFINEMENT** due Mar 10, 2022  
Priority: High
- ☒ **Alen Morres: ~~REVISITING MECHANICS AND FEATURE DEVELOPMENT~~** due Mar 10, 2022  
Priority: High
- ☒ **Keerthi T M: ~~DEVELOPING PROTOTYPE INTO COMMERCIAL PRODUCT~~** due Mar 20, 2022  
Priority: High
- ☒ **Keerthi T M: ~~FINAL TESTING AND REPORT GENERATION~~** due Mar 20, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: ~~REPORT AND DOCUMENT REVIEW~~** due Mar 22, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: ~~UPGRADING IDN PROTECC AND IDN DETECC WITH A MORE ROBUST REGEX~~** due Mar 27, 2022  
Priority: Medium

## ALPHA TESTING

- ☒ **Farhan P Mohammed: ~~TEST 1 - TEST DATA SET~~** due Mar 30, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: ~~SOFTWARE REVIEW AND WALKTHROUGH~~** due Mar 30, 2022  
Priority: Medium
- ☒ **Keerthi T M: ~~TEST 2 - FAIL TESTS AND VULNERABILITIES~~** due Apr 5, 2022  
Priority: Medium
- ☒ **Keerthi T M: ~~SECURITY REVIEW AND INSPECTION~~** due Apr 5, 2022  
Priority: Medium

## DEPLOYMENT STAGE

- ☒ **Alen Morres: ~~PORTING WEB EXTENSION~~** due Apr 15, 2022  
Priority: High
- ☒ **Alen Morres: ~~DEPLOYMENT IN WEB BROWSER~~** due Apr 15, 2022  
Priority: High

## BETA TESTING

- ☒ **Farhan P Mohammed: ~~BUGS~~** due Apr 18, 2022  
Priority: Medium
- ☒ **Farhan P Mohammed: ~~FEATURES REQUEST~~** due Apr 18, 2022  
Priority: Medium

## **WORK DONE : BROWSER EXTENSION (NOV 15th - APR 18th)**

As mentioned in the previous status report, MV3 was proposed to be an update to the way web-browsers handled extensions in the 4th quarter of 2021. We started developing extensions abiding to the MV3 standard. The change to MV3 brought in a lot of changes and brought in restrictions to the APIs previously used. We decided to get on with it, keeping future proofing in mind. In MV3, we were also not allowed to run the scripts persistently in the background, which was crucial for our extension.

But we were able to progress the mechanics of the extension by implementing several community aided workarounds in the Service workers and Content-Scripts.

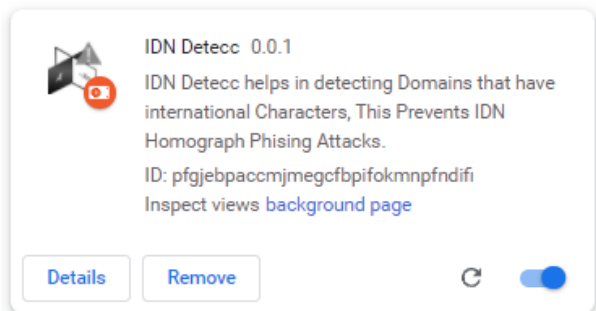
But when we were at the stage of using promises instead of call - backs for elements in scripts to communicate with each other, we discovered a chain of bugs for the features we were trying to implement. The complications that followed made the Service Worker behave unexpectedly, which made implementing and reviewing the changes to be inconsistent. Due to the undocumented Bugs in MV3 we decided to roll back to MV2 till chrome developers document the Bugs or provided Workarounds than relying on community aided workarounds which were still not efficacious in certain situations, And instead provide the necessary MV3 files to Port from MV2 to MV3 when a stable release of MV3 is available and continue development on MV2.

Developing the Mechanics in MV3 inspired us to change the way we approached the users with the extension. So in MV2 we decided to split the business model into two versions of the extensions that operated on different principles.

## Detailed Report:

### FIRST MODEL: IDN Detecc

- **IDN Detecc** - A minimally invasive extension that would detect any IDN website loaded and only notify the user if there is any IDN Detected.
  - If IDN detected - Pop up a notification for the user with the respective U-Labels and A-labels in it for the user to be aware.
  - The body of the extension displays the state of the website (IDN-Detected OR IDN-Not\_detected) for the user who have disabled their notifications.
  - Display the detection of IDN website by changing the icon in the extension bar.

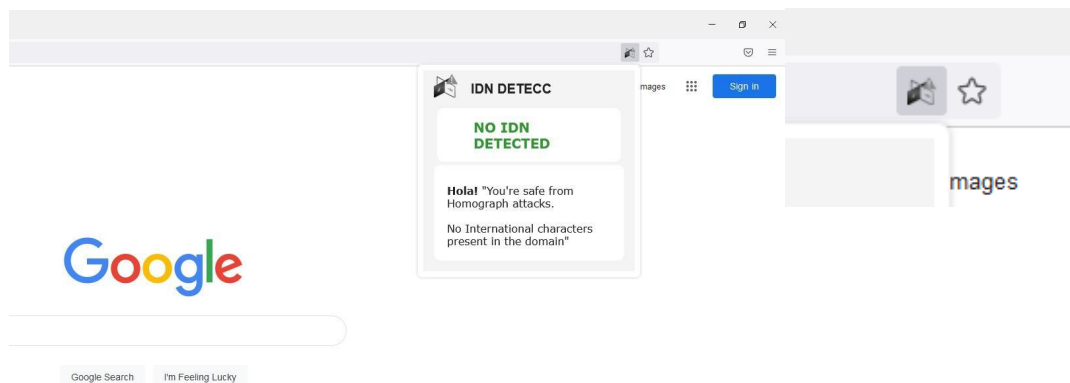


### Snippets of the extension in action:

#### IDN-Not Detected:

*Displays a dulled out icon on the extension bar.*

*Displays “NO IDN DETECTED” in the body of the extension when a user clicks on the extension.*

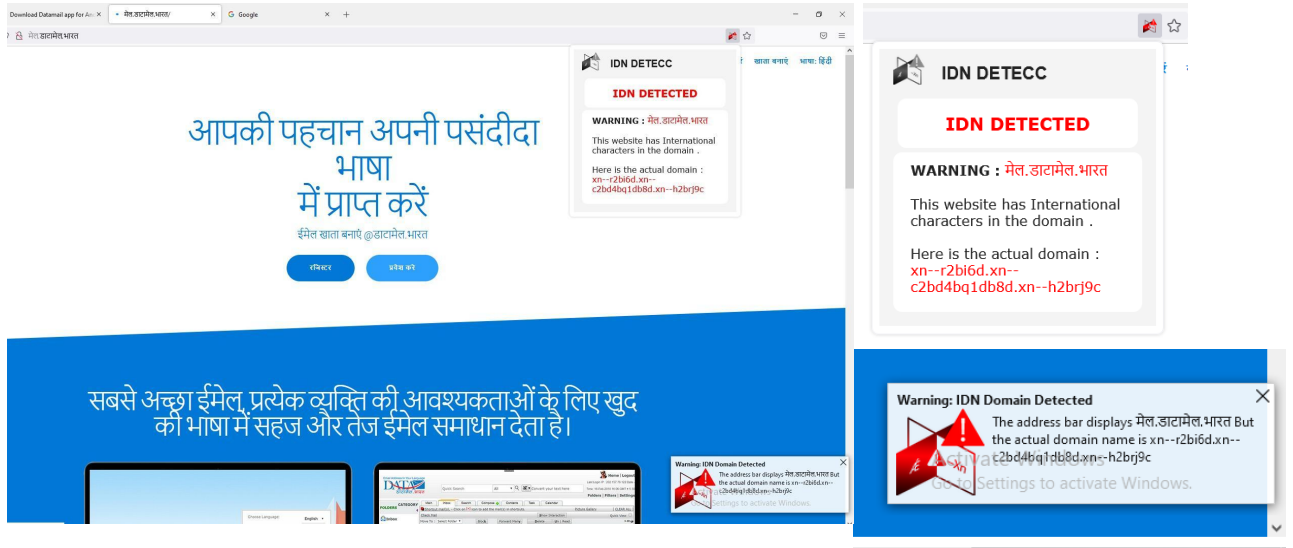


#### IDN Detected:

*Displays a red color caution icon on the extension bar.*

*A browser Notification pops up with a caution message that includes the A-label and U-label of the website visited.*

*Displays “IDN DETECTED” in the body of the extension when a user clicks on the extension.*



आपकी पहचान अपनी पसंदीदा  
भाषा  
में प्राप्त करें

ईमेल खाता बनाएँ @डाटा.मेल.भारत

रजिस्ट्रार

डोमेन

सबसे अच्छा ईमेल, प्रत्येक व्यक्ति की आवश्यकताओं के लिए खुद  
की भाषा में सहज और तेज ईमेल समाधान देता है।

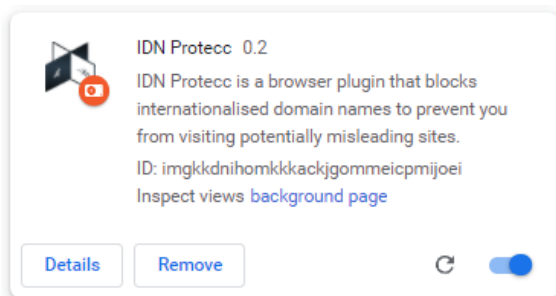


Warning: IDN Domain Detected  
The address bar displays मेल.डाटा.मेल.भारत But  
the actual domain name is xn--r2bi6d.xn--  
c2bd4bq1db8d.xn--h2brj9c  
Go to Settings to activate Windows.

Warning: IDN Domain Detected  
The address bar displays मेल.डाटा.मेल.भारत But  
the actual domain name is xn--r2bi6d.xn--  
c2bd4bq1db8d.xn--h2brj9c  
Go to Settings to activate Windows.

## SECOND MODEL: IDN Protecc

- **IDN Protecc**-Follows an invasive approach and blocks all IDN Websites from the user by default . And also provides additional options for users that want to customize the way websites were blocked, and options to allow the desired website that the user felt safe with.



- If IDN detected - Block the website from loading.
- Display the actual domain name of the website and provide users options to ,
  - Temporarily allow the domain
  - Add the domain to an exclusion list (White-List) and load the website again.
  - Display the whitelist and provide users options to,
    - Revoke temporarily allowed websites.
    - Remove from Whitelist.

### Screenshots of Extension in action :

#### IDN-Not-Detected:

*The normal icon is displayed on the extension bar.*

*Displays “No IDN domain blocked” in the body of the extension when a user clicks on the extension.*





## **IDN-Detected:**

*The icon in the extension bar changes to red.*

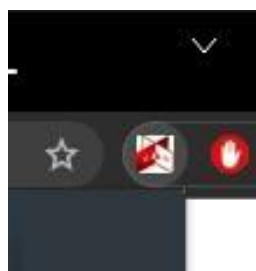
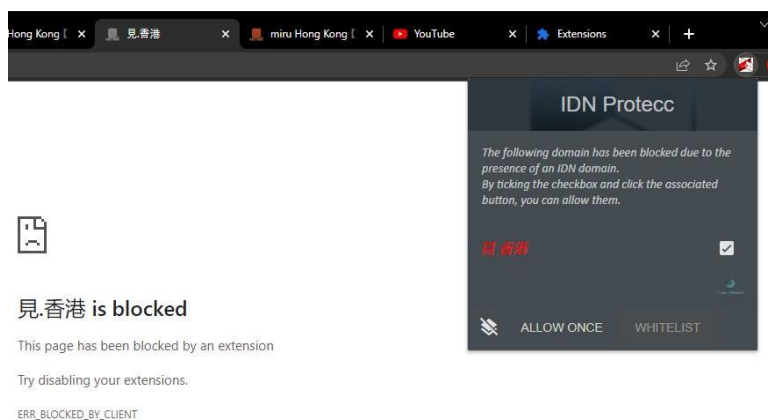
*Displays the U-Label of the website that is blocked in red color, along with a checkbox to select it.*

*The “Allow once” and “Whitelist” buttons are activated The user can click on either of these to,*

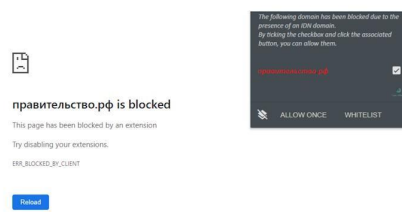
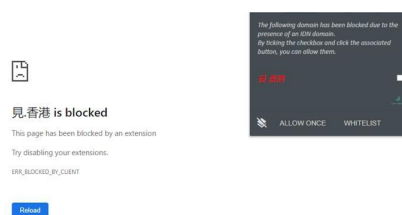
*-Allow the website temporarily instead of adding it to the whitelist. This will be reset if the user closes the browser and opens again, or if the user presses the revoke temporary button.*

*-Add the website to an extension list “Whitelist” and maintain the whitelist till the extension is reset or uninstalled .*

*-click on the whitelist icon to show the “whitelist”.*



## **Successfully identifying and blocking IDN Domains :**



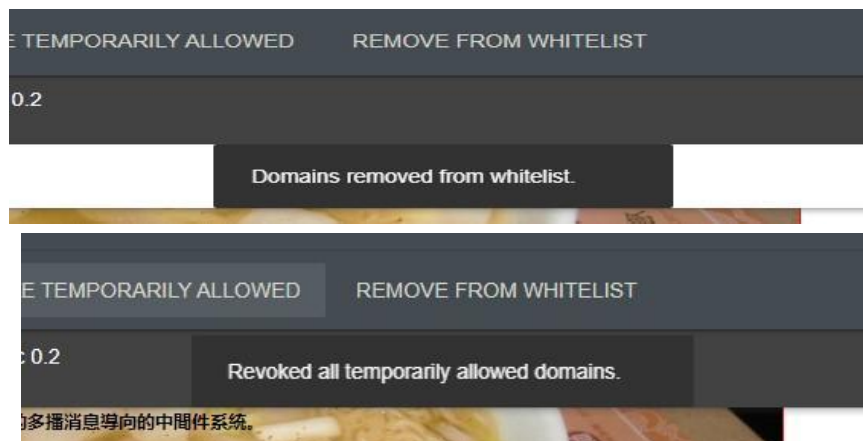
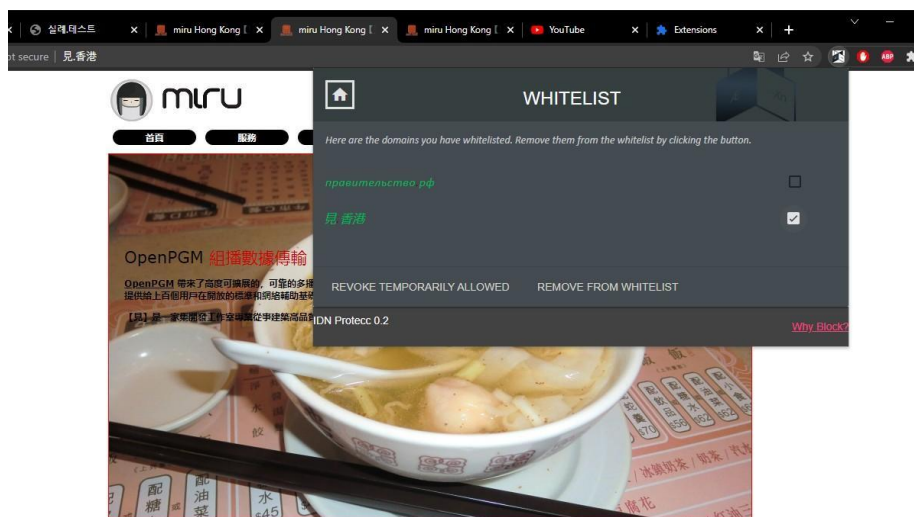
## Removing Whitelisted Domains :

The whitelisted domains are displayed in a list in the whitelist-page. The user can select the desired domain /domains and click on “Remove from whitelist” button to remove them from the White-list.

The ‘Revoke Temporarily Allowed’ revokes the websites that were allowed once temporarily.

The version of the extension is displayed on the bottom left.

A “Why block” hyperlink is provided at the bottom right, redirecting to a website about IDN homograph attacks.



## PROJECT TIMELINE : UA readiness of IDN Email Address in websites (April 20th - May 15th)

8/8/22, 11:47 PM

REGEX Library Development - Asana

### REGEX Library Development

Printed from Asana

#### IDEATION AND STUDY

- ☒ **Keerthi T M: RESEARCH AND KNOWLEDGE GATHERING** due Apr 20, 2022  
Priority: Low
- ☒ **Keerthi T M: BRAINSTORMING AND CONCEPTUALIZE** due Apr 21, 2022  
Priority: Low

#### DESIGN AND DEVELOPMENT

- ☒ **Farhan P Mohammed: TECHNICAL FLOW AND SKETCHING** due Apr 22, 2022  
Priority: Medium
- ☒ **Alen Morres: PROTOTYPE DESIGN** due Apr 23, 2022  
Priority: Medium
- ☒ **Keerthi T M: FEATURE IMPLEMENTATION** due Apr 27, 2022  
Priority: High
- ☒ **Keerthi T M: HTML5 INPUT FORM SUPPORT** due Apr 27, 2022  
Priority: High
- ☒ **Alen Morres: REGEX MECHANICS** due May 2, 2022  
Priority: High
- ☒ **Alen Morres: BI-DIRECTIONAL DOMAIN IDENTIFICATION** due May 8, 2022  
Priority: High

#### TESTING AND RECORD

- ☒ **Farhan P Mohammed: SYSTEM TESTING AND RECORD GENERATION** due May 10, 2022  
Priority: Medium
- ☒ **Alen Morres: DOCUMENTATION** due May 12, 2022  
Priority: Low
- ☒ **Keerthi T M: FINAL REVIEW** due May 16, 2022  
Priority: High

#### Backlog

- ☐ **N/A**  
Priority: Low

## WORK DONE :REGEX Library Development (APR 20th - MAY 15th)

### Objectives :

1. Identify ASCII only mail or IDN mail.
2. Identify and Invalidate emails with script-mixing. Test conditions for checking script-mixing
  - தமி @датамэйл.рф - invalid email address
  - ததாத @датамэйл.рф - invalid email address
  - தமி @வா .рф - invalid email address
3. Permitted identification of unicode@unicode.unicode email (क तिम)
4. Permit ascii@unicode.ascii / ascii@ascii.unicode / ascii@unicode.unicode unicode@ascii.ascii / unicode@unicode.ascii / unicode@ascii.unicode
5. Don't confuse valid Unicode domains with script mixed ones.

### Mechanics :

We had two thought process in validating the syntax of email with the mechanics of the library. The approaches are as follows :

#### 1) Unicode's character code identification :

- The principle is to compare character codes with the email string.
- Check condition for IDN email or ASCII only email.
- Scan for script mixing by comparing code of individual characters.
- The results are checked with REGEX for proper syntax of email.
- Unicode's character sets are updated and removed with new standards, this means the code has to be updated frequently to keep up with it.
- Observed unreliability for certain characters and takes more time to validate emails with more number of Unicode characters.
  - For example : -unicode@unicode.unicode

#### 2) Punycode(ACE) Identification :

- Use a punycode engine to identify (-xn prefix) and perform the conversions on Unicode characters.

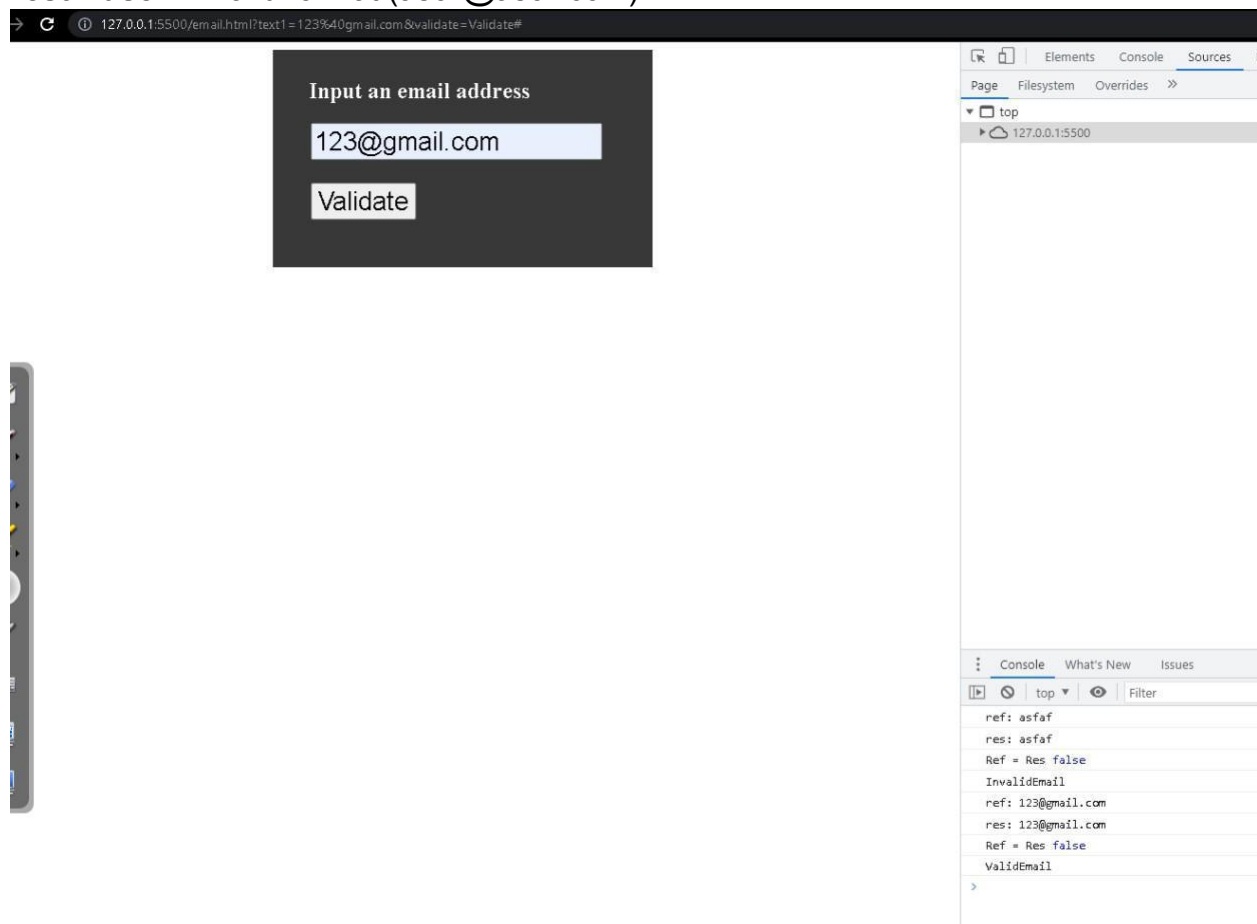
- If the converted email string has punycode, compare it with the IDN Syntax specific REGEX.
- Validate and display alerts for “IDN invalid email” state if invalid syntax or script mixing is detected. Display “IDN valid mail” state if syntactically matched.
- Display Valid and Invalid States for ASCII email

The involvement of punycode conversion at different parts adds a couple of seconds in validating the email, depending on the length of the Unicode characters present.

### Upcoming Tasks :

- Perform specific test cases to refine the REGEX and code for greater acceptance of unicode@unicode.unicode and bidirectional domains.
- Encapsulate the functions into wrappers and make it available as a syntax validation library.

### Test Case 1 : Valid format (ascii@ascii.com)



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:5500/email.html?text1=123%40gmail.com&validate=Validate#`. The main content area features a dark-themed input form with the text "Input an email address", a text input field containing "123@gmail.com", and a "Validate" button. To the right, the browser's developer tools are open, showing the Console tab. The console log contains the following entries:

```

ref: asfaf
res: asfaf
Ref = Res false
InvalidEmail
ref: 123@gmail.com
res: 123@gmail.com
Ref = Res false
ValidEmail

```

## Test Case 2 : Invalid email Format ASCII.

The screenshot shows a web browser at the URL `127.0.0.1:5500/email.html?text1=123%40gmail.com&validate=Validate#`. The application has a form titled "Input an email address" with a text input field containing "asfaf" and a "Validate" button. A tooltip message says: "Please include an '@' in the email address. 'asfaf' is missing an '@'." The browser's developer tools are open, showing the "Sources" tab with a file named `email.js`. The "Console" tab shows the following log:

```
ref: asfaf
res: asfaf
Ref = Res false
InvalidEmail
```

## Test Case 3 : Valid Format (ascii@unicode.unicode)

The screenshot shows the same web application as in Test Case 2, but with the email address `farhan@датамэйл.рф` entered in the input field. A tooltip message says: "Valid IDN email address!". The browser's developer tools are open, showing the "Sources" tab with a file named `email.js`. The "Console" tab shows the following log:

```
ref: farhan@m--80aakwke5bzfxn--plai
res: farhan@арамэйл.рф
Ref = Res true
IDN-ValidEmail
```

## Test Case 4 : Invalid email format (Ascii@unicode.unicode)

The screenshot shows a web browser at the URL `127.0.0.1:5500/email.html?text1=123%40gmail.com&validate=Validate#`. The application has a dark-themed input form with the text "Input an email address" and a text field containing `farhan@атамэйл...com`. A yellow "Validate" button is below the field. A modal dialog box is open, displaying the message "Invalid IDN email address!" with an "OK" button. The browser's developer tools are open, showing the "Console" tab with the following log entries:

```
ref: farhan@xn--80aakwke4bzf,xn--plai
res: farhan@asawasin.pq
Ref = Res true
IDN-ValidEmail
ref: farhan@xn--80aakwke4bzf@.com
res: farhan@asawasin@.com
Ref = Res true
IDN-InvalidEmail
```

The "Sources" tab shows the `emails.js` file with the following code:

```
var domain = inp
var ref = punycc
var res = punycc
console.log('ref')
console.log('res')
var validIdnRege
var validIdnRege =
if (ref=res){
console.log(
if (ref,
console.
alert("V
document
return t
} else {
console.
alert("I
document
return f
}
} else {
console.
```

## Test Case 5 : Valid Format (unicode@unicode.unicode)

The screenshot shows the same web application as in Test Case 4, but with a different input. The text field now contains `कीर्तिमोहन@डेटामेल.भारत`. The "Validate" button is still present. The modal dialog box is open, displaying the message "Invalid IDN email address!" with an "OK" button. The browser's developer tools are open, showing the "Console" tab with the following log entries:

```
ref: xn--11b4amc10dyb77hqa@xn--c2b4bq1d8d,xn--h2brj9c
res: कीर्तिमोहन@डेटामेल.भारत
Ref = Res true
IDN-InvalidEmail
```

The "Sources" tab shows the `emails.js` file with the following code:

```
var domain = inp
var ref = punycc
var res = punycc
console.log('ref')
console.log('res')
var validIdnRege
var validIdnRege =
if (ref=res){
console.log(
if (ref,
console.
alert("V
document
return t
} else {
console.
alert("I
document
return f
}
} else {
console.
```

**CONCLUSION :**

The project describes the results of the suggested solution put into practise to validate the legitimacy of IDN domains and Email Acceptance in client side. This implementation demonstrated that UA and EA in the current system may be enhanced with relatively few changes to the current system and developing an easy to replicate solution.