

# ASSIGNMENT COVERSHEET



**Murdoch**  
UNIVERSITY

<b>Student Number:</b>	35629196
Surname:	Vijaya Raghavan
Given name:	Maheyshvar
Email:	35629196@student.murdoch.edu.au

<b>Unit Code:</b>	ICT171
Unit name:	Introduction to Server Environments and Architectures
Enrolment mode:	External (UCSI College, Kuala Lumpur Campus)
Date:	6 <sup>th</sup> April 2025
Assignment name:	Cloud Project & Video Explainer
Tutor:	Ts. Soo Weng Jyh (Jason)
Unit Coordinator:	Dr David Murray

## Student's Declaration:

- Except where indicated, the work I am submitting in this assignment is my own work and has not been submitted for assessment in another unit.
- This submission complies with Murdoch University's academic integrity commitments. I am aware that information about plagiarism and associated penalties can be found at <http://our.murdoch.edu.au/Educational-technologies/Academic-integrity/>. If I have any doubts or queries about this, I am further aware that I can contact my Unit Coordinator prior to submitting the assignment.
- I acknowledge that the assessor of this assignment may, for the purpose of assessing this assignment:
  - reproduce this assignment and provide a copy to another academic staff member; and/or
  - submit a copy of this assignment to a plagiarism-checking service. This web-based service may retain a copy of this work for the sole purpose of subsequent plagiarism checking, but has a legal agreement with the University that it will not share or reproduce it in any form.
- I have retained a copy of this assignment.
- I will retain a copy of the notification of receipt of this assignment. If you have not received a receipt within three days, please check with your Unit Coordinator.

I am aware that I am making this declaration by submitting this document electronically and by using my Murdoch ID and password it is deemed equivalent to executing this declaration with my written signature.

**Optional Comments to Tutor:**

None.

## Table of Content

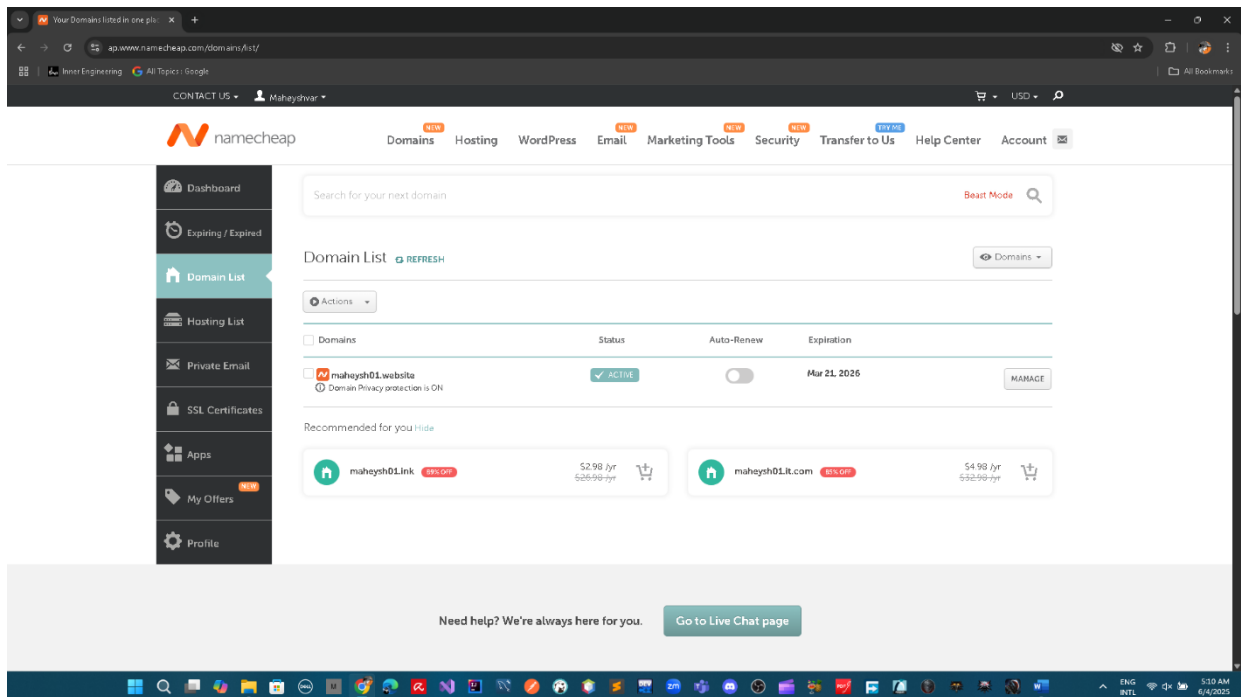
<b>Cover Page.....</b>	<b>I</b>
<b>Table of Content.....</b>	<b>III</b>
<b>Introduction.....</b>	<b>1</b>
<b>DNS Documentation.....</b>	<b>2</b>
<b>SSL/TLS Documentation.....</b>	<b>5</b>
<b>Script Documentation.....</b>	<b>8</b>
Navigation Panel.....	8
Comments Feature.....	12
<b>Github Link.....</b>	<b>14</b>
<b>Link to Web Server.....</b>	<b>14</b>
<b>Link to Video Explainer.....</b>	<b>14</b>
<b>References.....</b>	<b>15</b>

## **Introduction**

Literature Haven is a book lovers' website offering an engaging literary discussion medium. With a user-friendly interface and easy-to-navigate design, the website caters to readers of every genre, from classical literature to modern fiction. The primary objective of Literature Haven is to create a successful literary society whereby readers can find new books, read about genres, and engage in meaningful discussions. The website contains detailed book reviews, themed reading lists, and a blog section where readers can share observations, criticisms, and recommendations. Through these, Literature Haven encourages the culture of reading and intellectual discourse.

This report investigates crucial technical aspects implemented during this website's development, such as the site's DNS configuration, SSL integration, and script documentation. These factors are essential to the platform's security, stability, and maintainability.

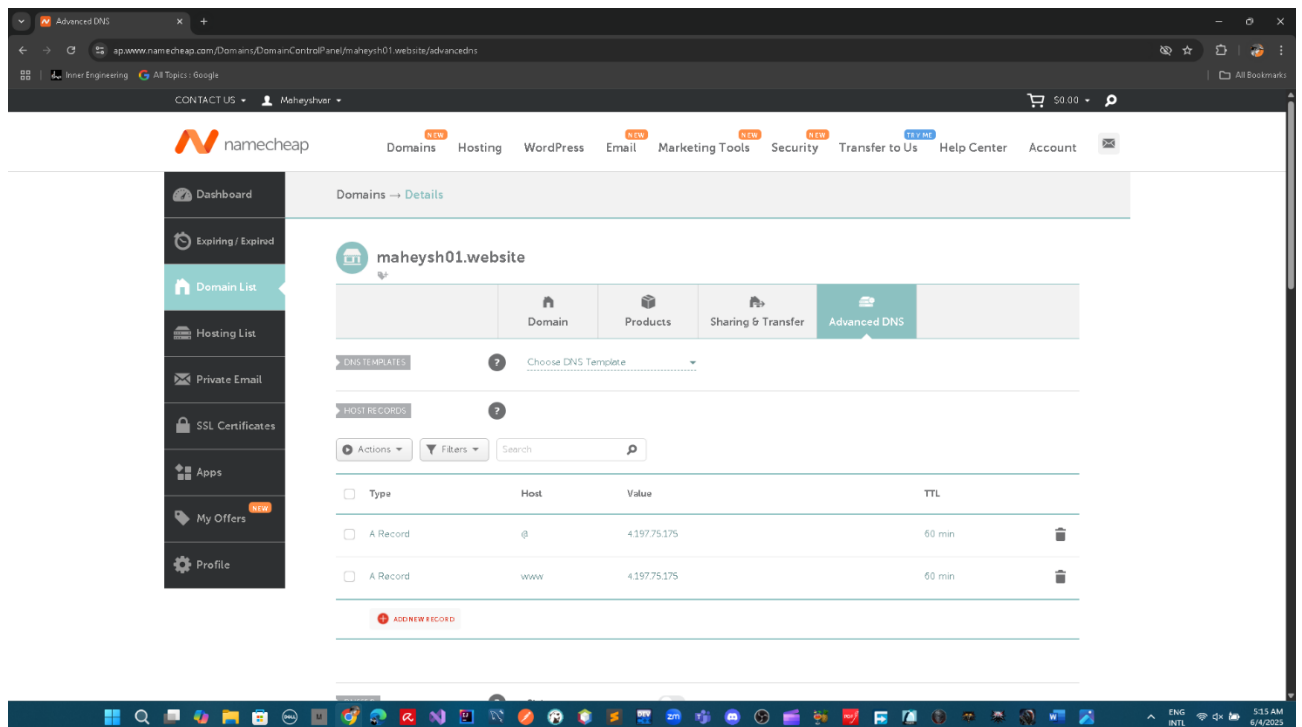
## DNS Documentation



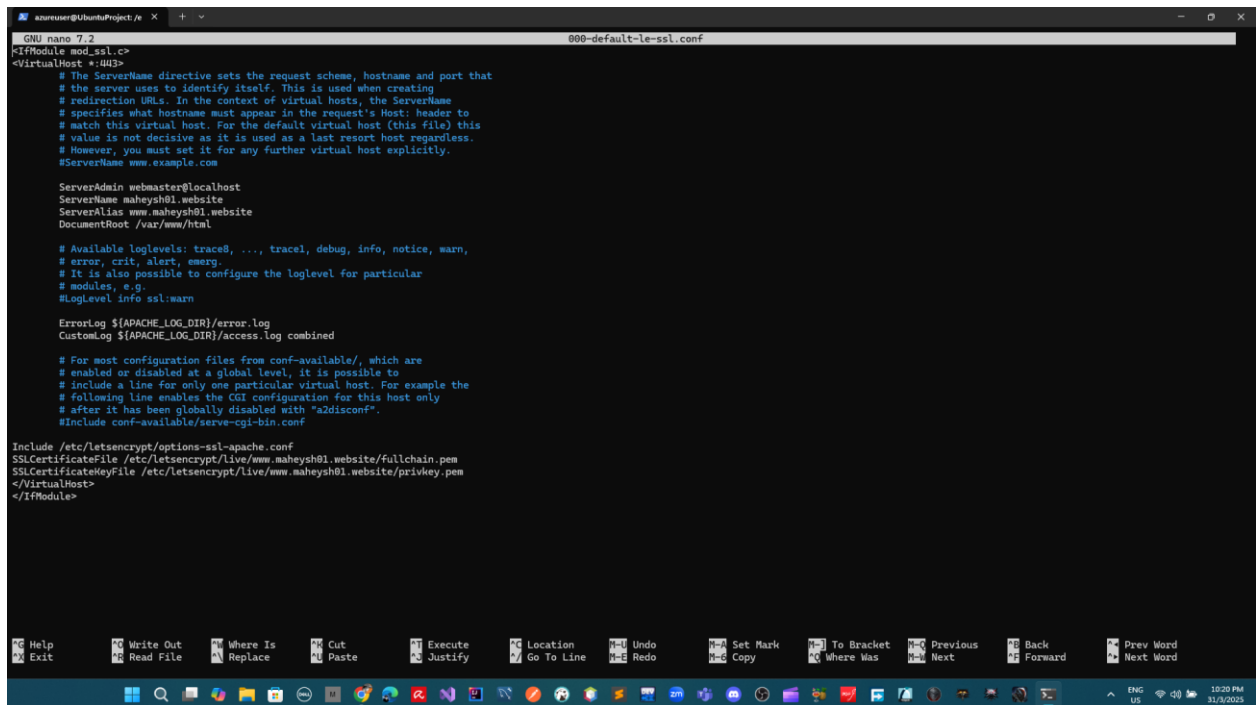
The domain name, maheysh01.website, was purchased through a known domain and hosting selling company called NameCheap for 0.99 USD, serving as the website's identity. It was then incorporated into the website through PowerShell commands, which allowed for a smooth process of adding the domain onto the web server. Configuration involved altering the Domain Name System (DNS) records so the traffic would be routed properly to the given hosting environment.

The process began by logging into NameCheap's Advanced DNS panel, where A (Address) records were created based on the needs of hosting. The A records were pointed to the website's IP address so that the users can access the website via the

registered domain name.



Apache web server configuration was also modified through PowerShell for proper redirection and domain management. The necessary configurations were applied remotely to the server through PowerShell, simplifying the process. ServerName directive was also set as **maheysh01.website**, and a ServerAlias for **www.maheysh01.website** was established to have equal access to both versions of the domain. The Apache configuration file, accessed at **/etc/apache2/sites-available/000-default.conf**, was modified to open and edit the file remotely.



```
GNU nano 7.2                                000-default-le-ssl.conf
<IfModule mod_ssl.c>
<VirtualHost *:443>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    ServerName maheysh01.website
    ServerAlias www.maheysh01.website
    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

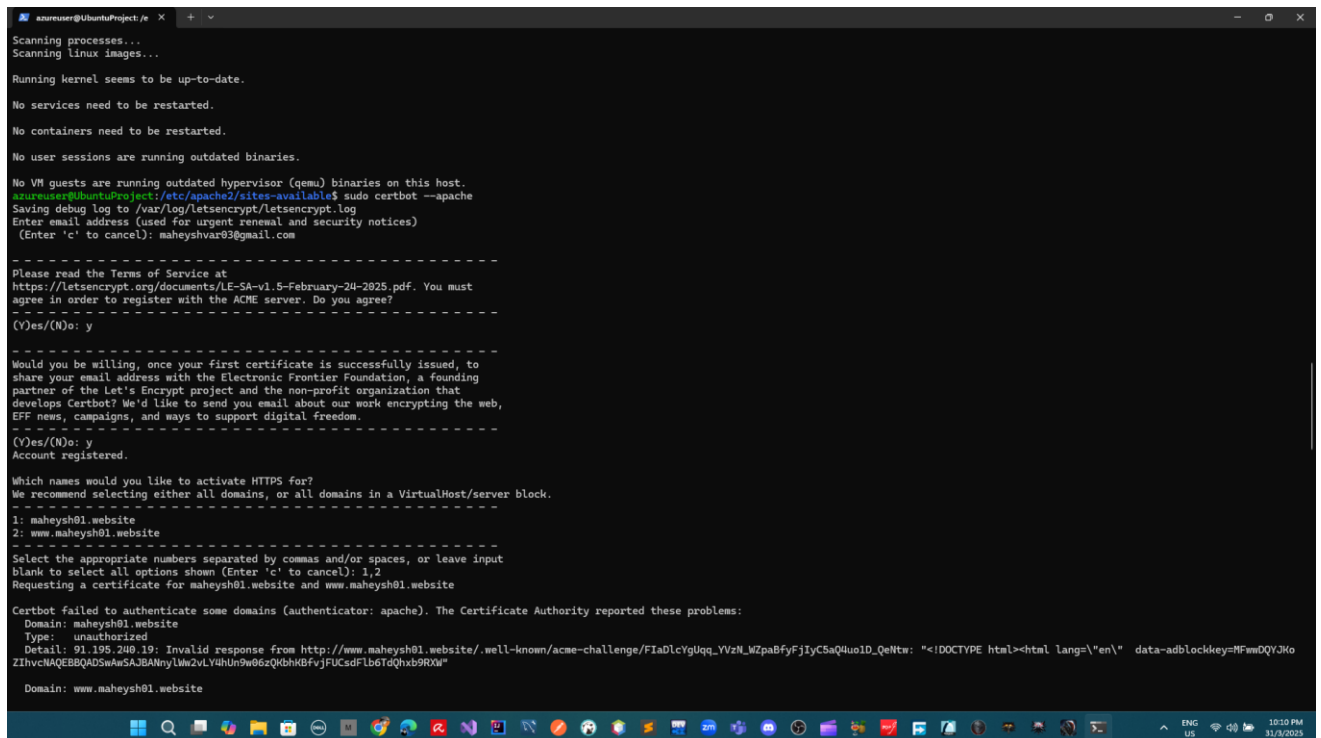
    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    Include /etc/letsencrypt/options-ssl-apache.conf
    SSLCertificateFile /etc/letsencrypt/live/www.maheysh01.website/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/www.maheysh01.website/privkey.pem
</VirtualHost>
</IfModule>
```

Through this structured DNS setup, maheysh01.website was effectively linked to the hosting environment and made available while domain resolution is properly executed. This setup enhances the reliability of the website since it can be easily accessed while a smooth and secure process of domain management is maintained. (Sy, 2024)

## SSL/TLS Documentation

Securing a website with HTTPS is essential for encrypting data, building user trust, and compliance with modern web security best practices. (Palii, 2023) SSL was configured for this site using Apache and Let's Encrypt in the process, enabling secure communication between the server and users.



```
anuruser@UbuntuProject: ~$ sudo certbot --apache
Scanning processes...
Scanning Linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): maheyshvar03@gmail.com

-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.5-February-24-2025.pdf. You must
agree in order to register with the ACME server. Do you agree?
-----
(Y)es/(N)o: y

-----
Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----
(Y)es/(N)o: y
Account registered.

Which names would you like to activate HTTPS for?
We recommend selecting either all domains, or all domains in a VirtualHost/server block.
-----
1: maheysh01.website
2: www.maheysh01.website
-----
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 1,2
Requesting a certificate for maheysh01.website and www.maheysh01.website

Certbot failed to authenticate some domains (authenticator: apache). The Certificate Authority reported these problems:
Domain: maheysh01.website
Type: unauthorized
Detail: 91.195.248.19: Invalid response from http://www.maheysh01.website/.well-known/acme-challenge/F1aDlCvUqQ_YVzN_WZpaBfyFjIyC5aQ4uo1D_QeHtw: "<!DOCTYPE html><html lang='en'" data-adblockkey=MFwDQYJKo
ZihvcNAQEBAQ0SAmSAJBANmyLWwZuLY4hUn9w06zQhkhBFvjFUCsdfLb6TdQhxb9RXW"
```

The first step was to allow SSL on the Apache web server. This was achieved using the **a2enmod ssl** command to allow the necessary modules so that Apache could handle secure connections. The default SSL configuration was made available using **a2ensite default-ssl.conf** and restarted the Apache service so that the changes would be implemented.

A free SSL certificate was then obtained by utilizing a tool provided by Let's Encrypt called Certbot. Certbot simplifies the process by automating the request for the certificate, installation of the certificate, and renewal. The request for the certificate was initiated by executing the command **certbot --apache**, during which the domain



was selected to be encrypted. SSL files such as fullchain.pem and privkey.pem were automatically created by Certbot. These files were then saved securely on the server.

Once the certificate was successfully issued, Certbot automatically configured Apache to use the certificate, modifying the necessary virtual host files. The certificate was implemented, ensuring HTTPS was in effect on the site. This setting encrypts all traffic to the website so that it cannot be intercepted and overall security is improved.

```
Address: 162.255.119.191
azuruser@UbuntuProject: /etc/apache2/sites-available$ sudo certbot --apache
Saving debug log to /var/log/letsencrypt/letsencrypt.log

Which names would you like to activate HTTPS for?
We recommend selecting either all domains, or all domains in a VirtualHost/server block.
-----
1: maheysh01.website
2: www.maheysh01.website
-----
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 1
Requesting a certificate for maheysh01.website

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/maheysh01.website/fullchain.pem
Key is saved at: /etc/letsencrypt/live/maheysh01.website/privkey.pem
This certificate expires on 2025-06-29.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate
Successfully deployed certificate for maheysh01.website to /etc/apache2/sites-available/000-default-le-ssl.conf
Congratulations! You have successfully enabled HTTPS on https://maheysh01.website

-----
If you like Certbot, please consider supporting our work by:
 * Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
 * Donating to EFF: https://eff.org/donate-le
-----
azuruser@UbuntuProject: /etc/apache2/sites-available$ sudo a2enmod ssl
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
azuruser@UbuntuProject: /etc/apache2/sites-available$ sudo a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
systemctl reload apache2
azuruser@UbuntuProject: /etc/apache2/sites-available$ sudo systemctl reload apache2
azuruser@UbuntuProject: /etc/apache2/sites-available$ sudo certbot --apache
Saving debug log to /var/log/letsencrypt/letsencrypt.log

Which names would you like to activate HTTPS for?
We recommend selecting either all domains, or all domains in a VirtualHost/server block.
-----
1: maheysh01.website
2: www.maheysh01.website
-----
```

```
azureuser@UbuntuProject: /e
-----
azureuser@UbuntuProject:/etc/apache2/sites-available$ sudo a2enmod ssl
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
azureuser@UbuntuProject:/etc/apache2/sites-available$ sudo a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
azureuser@UbuntuProject:/etc/apache2/sites-available$ sudo systemctl reload apache2
azureuser@UbuntuProject:/etc/apache2/sites-available$ sudo certbot --apache
Saving debug log to /var/log/letsencrypt/letsencrypt.log

Which names would you like to activate HTTPS for?
We recommend selecting either all domains, or all domains in a VirtualHost/server block.
-----
1: maheys01.website
2: www.maheys01.website
-----
Select the appropriate numbers separated by commas and/or spaces, or leave input
blank to select all options shown (Enter 'c' to cancel): 2
Requesting a certificate for www.maheys01.website

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/www.maheys01.website/fullchain.pem
Key is saved at: /etc/letsencrypt/live/www.maheys01.website/privkey.pem
This certificate expires on 2025-06-29.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate
Successfully deployed certificate for www.maheys01.website to /etc/apache2/sites-enabled/000-default-le-ssl.conf
Congratulations! You have successfully enabled HTTPS on https://www.maheys01.website

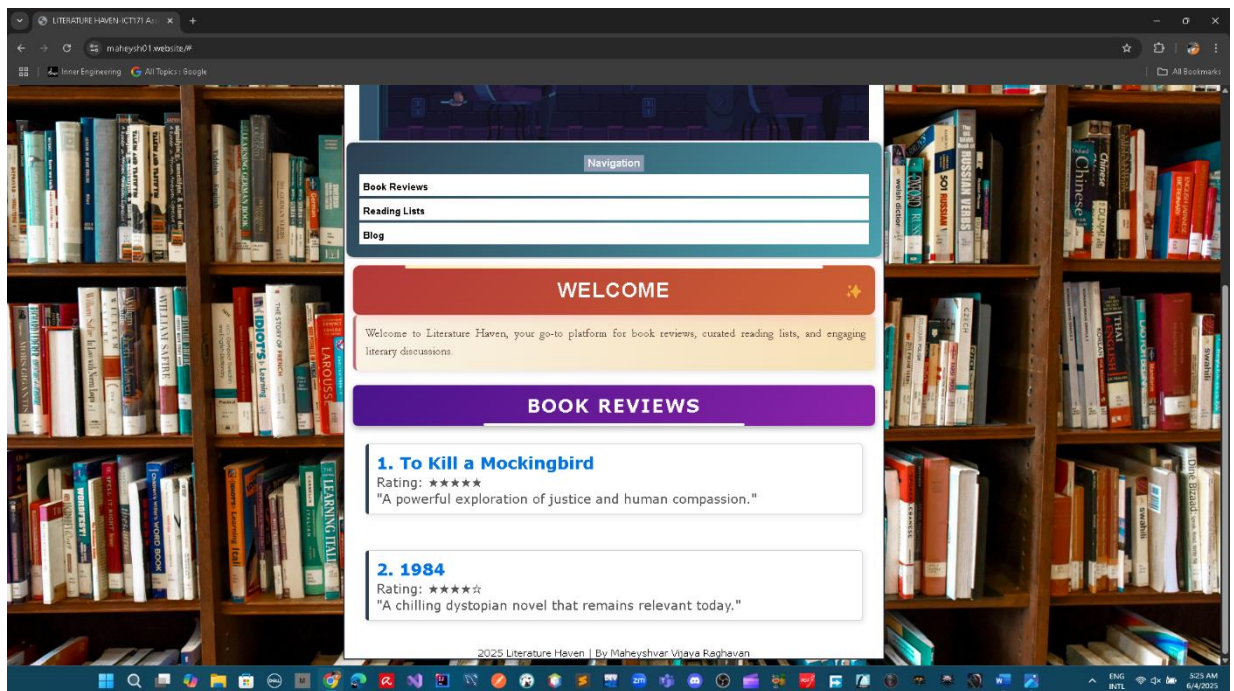
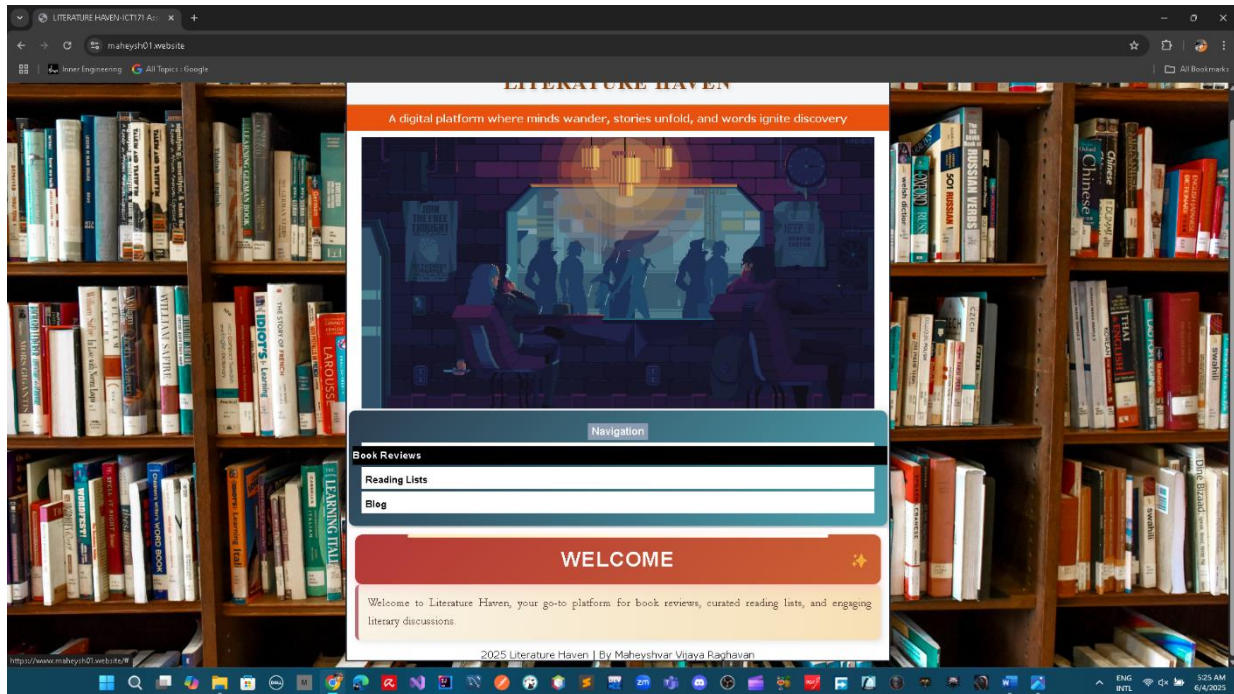
-----
If you like Certbot, please consider supporting our work by:
  * Donating to ISRG / Let's Encrypt:  https://letsencrypt.org/donate
  * Donating to EFF:  https://eff.org/donate-le
-----
azureuser@UbuntuProject:/etc/apache2/sites-available$ ls -la
total 20
drwxr-xr-x 2 root root 4096 Mar 31 14:06
drwxr-xr-x 8 root root 4096 Mar 31 14:06 .
-rw-r--r-- 1 root root 1604 Mar 31 14:06 000-default-le-ssl.conf
-rw-r--r-- 1 root root 1555 Mar 31 13:44 000-default.conf
-rw-r--r-- 1 root root 4572 Mar 18 2020 default-ssl.conf
azureuser@UbuntuProject:/etc/apache2/sites-available$
```

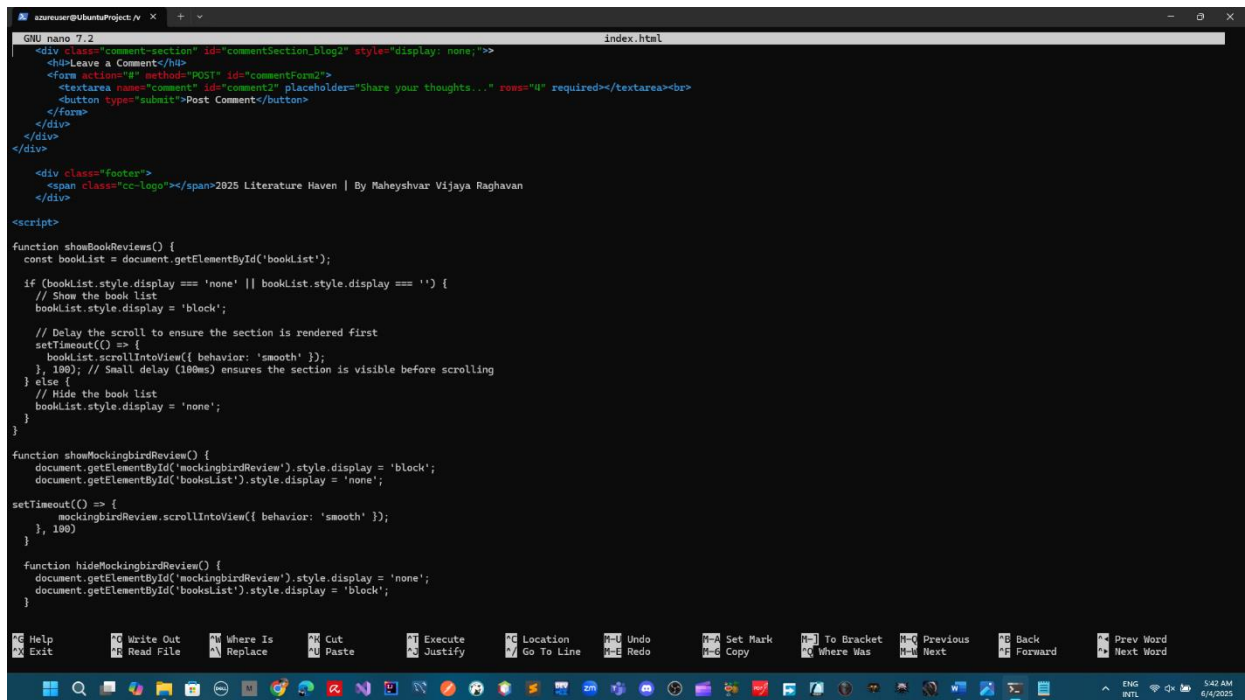
Available configurations were listed in the /etc/apache2/sites-available directory , and SSL configurations were confirmed to be available. With this, the website is now secure and accessible through HTTPS, which makes it more credible and secure for user interactions.

This process secures the website and ensures compliance with modern web security standards, providing users with a safe and trustworthy browsing experience. (Palii, 2023)

## Script Documentation

### Navigation Panel





```
GNU nano 7.2 index.html
<div class="comment-section" id="commentSection_blog1" style="display: none;">
  <h3>Leave a Comment</h3>
  <form action="" method="POST" id="commentForm2">
    <textarea name="comment" id="comment2" placeholder="Share your thoughts..." rows="4" required></textarea><br>
    <button type="submit">Post Comment</button>
  </form>
</div>

<div class="footer">
  <span class="cc-logo"></span>2025 Literature Haven | By Maheysvar Vijaya Raghavan
</div>

<script>
function showBookReviews() {
  const bookList = document.getElementById('bookList');

  if (bookList.style.display === 'none' || bookList.style.display === '') {
    // Show the book list
    bookList.style.display = 'block';

    // Delay the scroll to ensure the section is rendered first
    setTimeout(() => {
      bookList.scrollIntoView({ behavior: 'smooth' });
    }, 100); // Small delay (100ms) ensures the section is visible before scrolling
  } else {
    // Hide the book list
    bookList.style.display = 'none';
  }
}

function showMockingbirdReview() {
  document.getElementById('mockingbirdReview').style.display = 'block';
  document.getElementById('bookList').style.display = 'none';

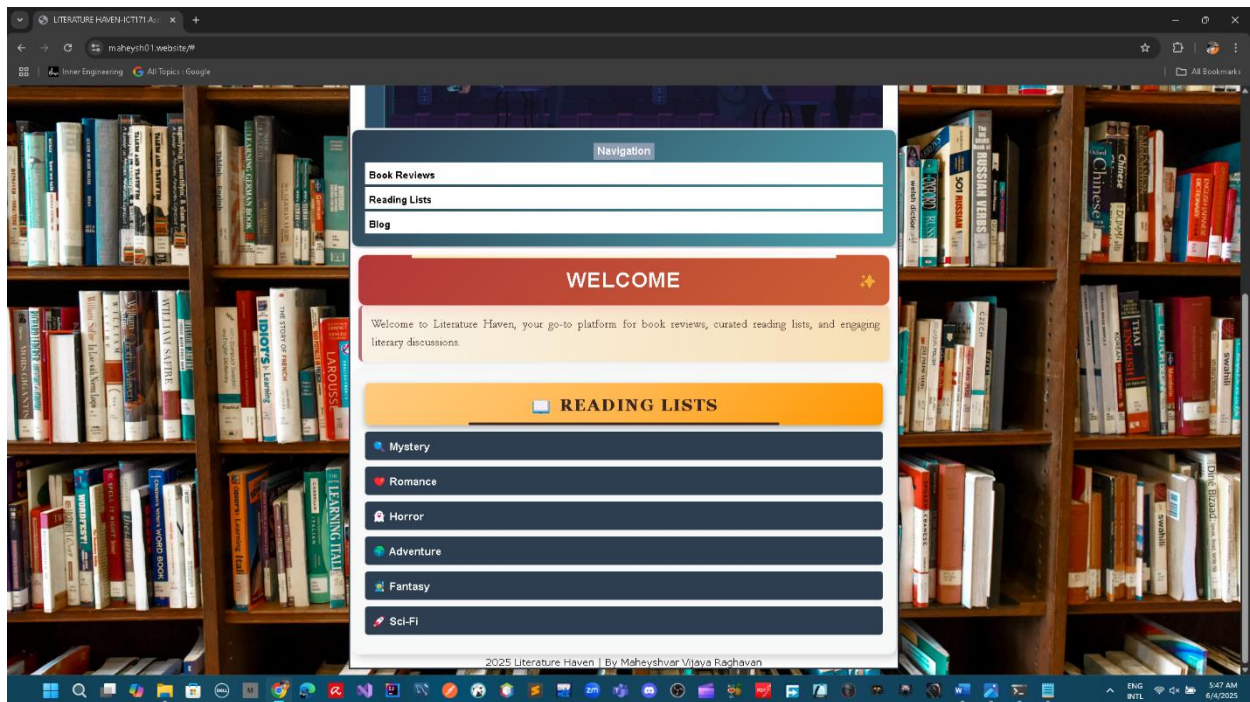
  setTimeout(() => {
    mockingbirdReview.scrollIntoView({ behavior: 'smooth' });
  }, 100)
}

function hideMockingbirdReview() {
  document.getElementById('mockingbirdReview').style.display = 'none';
  document.getElementById('bookList').style.display = 'block';
}
}
```

JavaScript functions for this website were implemented to enhance user interaction, allowing for dynamically managing the visibility of sections such as book reviews, reading lists, and blogs. It offers a smooth switch between them, thus simplifying user experience. The book review section on the website toggles the visibility of book lists and reviews for every book.

The `showBookReviews()` function reveals the book list and smoothly scrolls to the section with a brief delay to ensure visibility. Similarly, `showMockingbirdReview()` and `showReview1984()` display reviews for *To Kill a Mockingbird* and *1984*, respectively, while hiding the main book list. Corresponding `hideMockingbirdReview()` and `hideReview1984()` functions restore the book list when users close a review.





```

GNU nano 7.2 index.html
document.getElementById("review1984").style.display = "block"; // Show the review
document.getElementById("bookssList").style.display = "none"; // Hide main book list

setTimeout(() => {
  review1984.scrollIntoView({ behavior: 'smooth' });
}, 100)

function hideReview1984() {
  document.getElementById("review1984").style.display = "none"; // Hide the review
  document.getElementById("bookssList").style.display = "block"; // Show main book list
}

function showReadingLists() {
  const readingList = document.getElementById('readingList');

  if (readingList.style.display === 'none' || readingList.style.display === '') {
    // Show the reading list
    readingList.style.display = 'block';

    // Delay the scroll to ensure the section is rendered first
    setTimeout(() => {
      readingList.scrollIntoView({ behavior: 'smooth' });
    }, 100); // Small delay (100ms) ensures the section is visible before scrolling
  } else {
    // Hide the reading list
    readingList.style.display = 'none';
  }
}

document.addEventListener("DOMContentLoaded", function () {
  let bookLists = document.querySelectorAll("book-list");
  bookLists.forEach(list => list.style.display = "none");
});

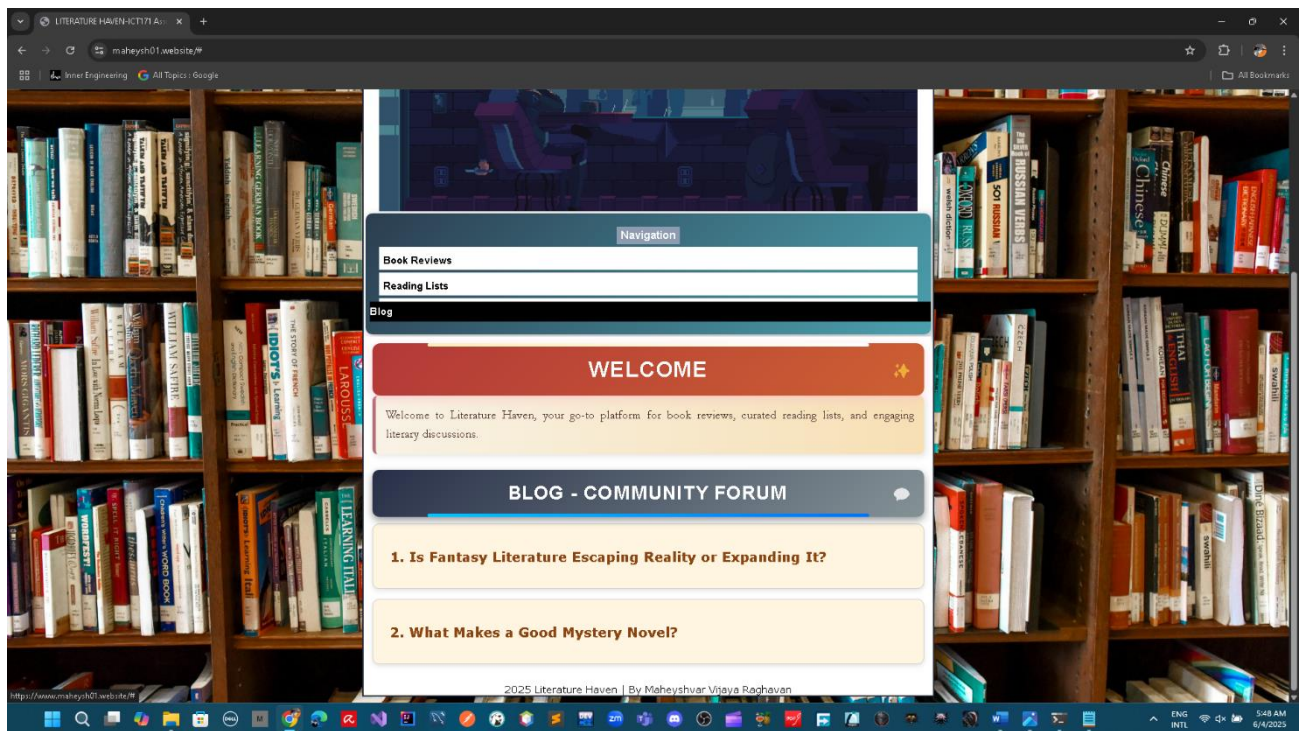
function toggleGenre(genreId) {
  let list = document.getElementById(genreId);
  if (list.style.display === "none" || list.style.display === "") {
    list.style.display = "block";
  } else {
    list.style.display = "none";
  }
}

function showBlog() {
  const blogSection = document.getElementById('blogSection');
}

```

The reading list part also contains the same toggle function. The function `showReadingLists()` displays or hides the reading list and also contains a smooth scrolling function to provide a smooth transition. The genre filtering method, `toggleGenre(genreId)`, is implemented to enable users to expand or collapse lists of

books according to their genres by dynamically displaying or hiding content. This functionality offers a neat browsing experience.



```
GNU nano 7.2 index.html
if (readingList.style.display === 'none' || readingList.style.display === '') {
  // Show the reading list
  readingList.style.display = 'block';

  // Delay the scroll to ensure the section is rendered first
  setTimeout(() => {
    readingList.scrollToView({ behavior: 'smooth' });
  }, 100); // Small delay (100ms) ensures the section is visible before scrolling
} else {
  // Hide the reading list
  readingList.style.display = 'none';
}

document.addEventListener("DOMContentLoaded", function () {
  let bookLists = document.querySelectorAll("book-list");
  bookLists.forEach(list => list.style.display = "none");
});

function toggleGenre(genreId) {
  let list = document.getElementById(genreId);
  if (list.style.display === "none" || list.style.display === "") {
    list.style.display = "block";
  } else {
    list.style.display = "none";
  }
}

function showBlog() {
  const blogSection = document.getElementById('blogSection');
  if (blogSection.style.display === 'none' || blogSection.style.display === '') {
    // Show the Blog section
    blogSection.style.display = 'block';

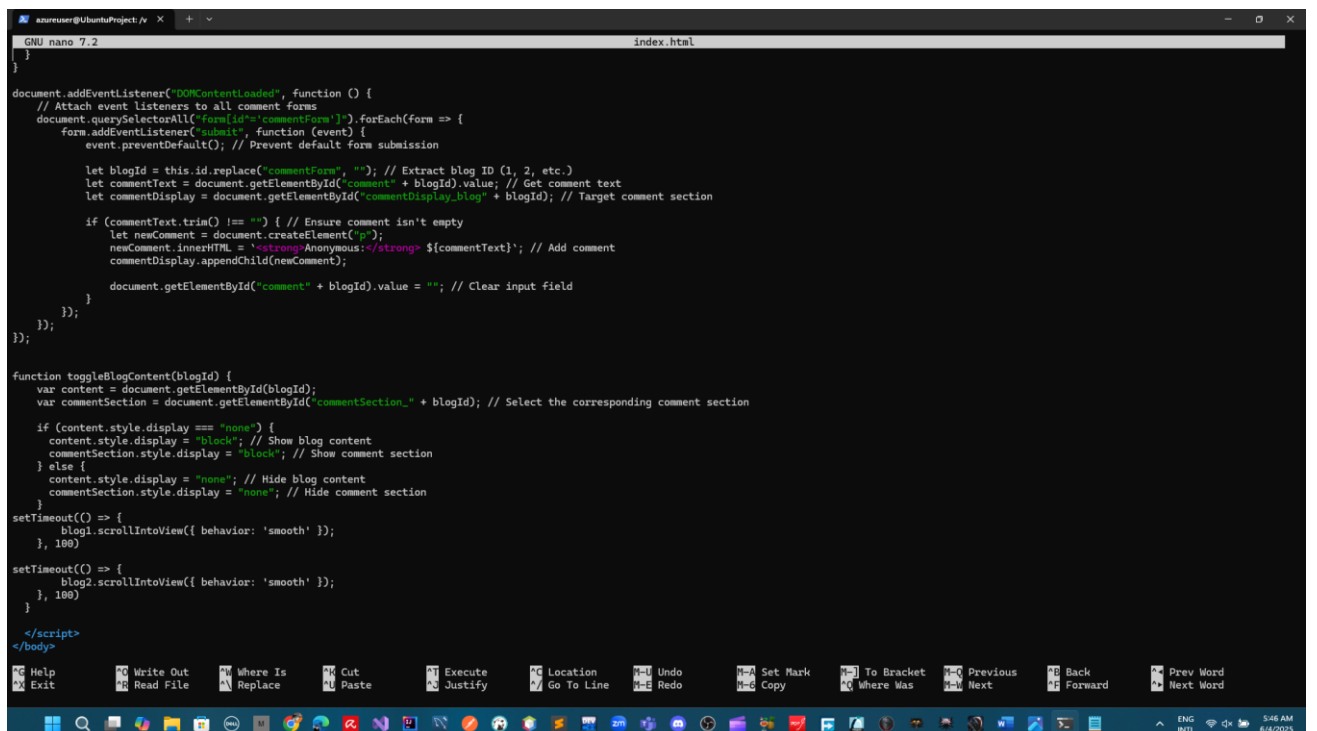
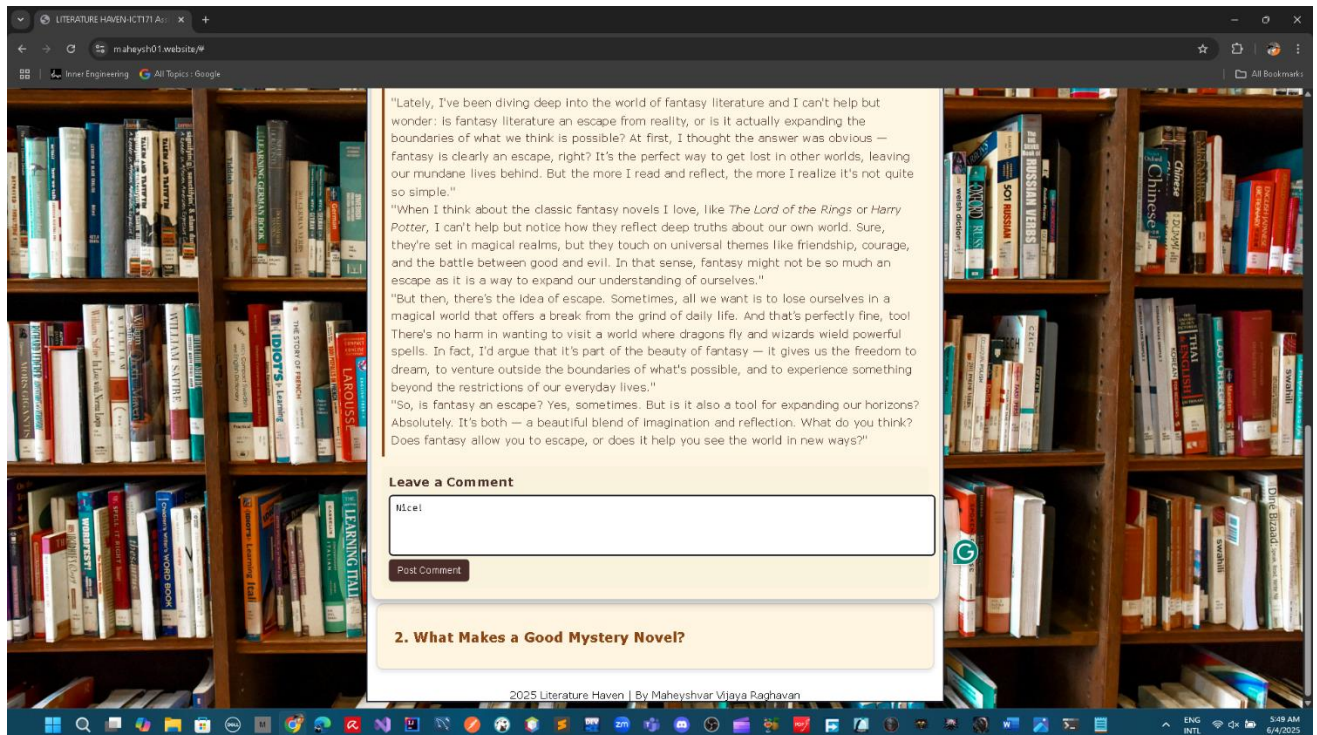
    // Delay the scroll to ensure smooth transition
    setTimeout(() => {
      blogSection.scrollToView({ behavior: 'smooth' });
    }, 100); // Small delay ensures the section is visible before scrolling
  } else {
    // Hide the Blog section
    blogSection.style.display = 'none';
  }
}

document.addEventListener("DOMContentLoaded", function () {
  // Attach event listeners to all comment forms
```

For the blog section, showBlog() controls whether the blog posts are shown or not, allowing a fluid scroll transition whenever users look at the area. There is an

interactive toggle feature for each blog post, `toggleBlogContent(blogId)`, which opens or closes its content and also its corresponding comment area.

## Commenting System



The commenting system is also handled dynamically. Upon loading the page, the event listener listens for all types of comments to allow for smooth interaction. When submitted, the script prevents reloading of the page, retrieves the blog ID, reads the comment text, and submits it below the related blog post while ensuring that no comments with nothing in them are submitted. There's just a little tweak to be done on `newComment.innerHTML`, such that `${commentText}` would be set within backticks as correct template literal syntax.

For optimization of the website's initial layout, a `DOMContentLoaded` event is used to keep all book lists defaulted as hidden and only open upon a user starting interaction with them. This enhances the loading process by avoiding extra components from opening at launch.



**Github Link**

[https://github.com/Maheysh01/ICT171\\_Assignment-2.git](https://github.com/Maheysh01/ICT171_Assignment-2.git)

**Web Link to Server**

<https://www.maheysh01.website/>

**Link to Video Explainer**

<https://murdochuniversity->

[my.sharepoint.com/personal/75001272\\_murdoch\\_edu\\_au/\\_layouts/15/stream.aspx?id=%2Fpersonal%2F75001272%5Fmurdoch%5Fedu%5Fau%2FDocuments%2FICT171%20%2D%20Jan%20Trimester%202025%2FICT171%20Maheyshvar%20%2D%20Video%20Explainer%2Emp4&referrer=StreamWebApp%2EWeb&referrerScenario=AddressBarCopied%2Eview%2E140a185a%2Df447%2D40aa%2Db604%2D4e7f4e0fe620](https://my.sharepoint.com/personal/75001272_murdoch_edu_au/_layouts/15/stream.aspx?id=%2Fpersonal%2F75001272%5Fmurdoch%5Fedu%5Fau%2FDocuments%2FICT171%20%2D%20Jan%20Trimester%202025%2FICT171%20Maheyshvar%20%2D%20Video%20Explainer%2Emp4&referrer=StreamWebApp%2EWeb&referrerScenario=AddressBarCopied%2Eview%2E140a185a%2Df447%2D40aa%2Db604%2D4e7f4e0fe620)

## References

- Palii, I. (3 October, 2023). *What is HTTPS and Why is it Essential for Web Security?* Retrieved from Sitechecker: <https://sitechecker.pro/what-is-https/#:~:text=HTTPS%20works%20by%20using%20a,connecting%20to%20the%20correct%20website.>
- Sy, C. (29 July, 2024). *Domain Name Management: Everything You Need to Know.* Retrieved from Domain.com: <https://www.domain.com/blog/domain-management-101/>