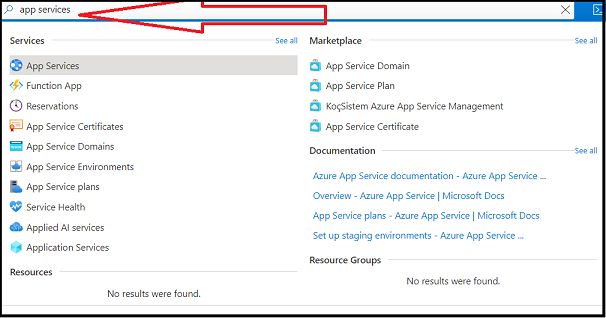
**Project 1: Security Cloud Application**

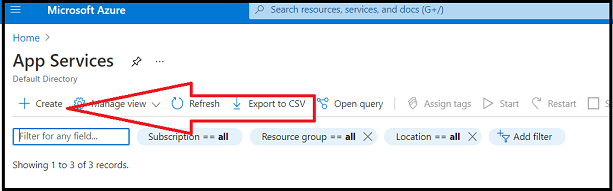
Day 1: Build, Host and Design web application

Part 1: Azure Web App

1. Logged in to Azure personal account.
2. Select “APP SERVICES” from search.



1. Select “+ Create” to create your application.



1. Fill up “Basics” tab as following

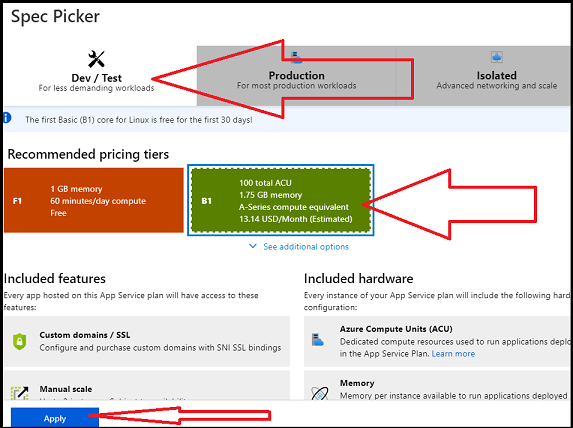
* Subscription/Resource Group: CloudWebApp
* Name: RipalPatel
* Publish: Select "Code."
* Runtime Stack: Select "PHP 7.4."
* Operating System: Select "Linux."
* Region: Canada Central

Graphical user interface, text, application, email

Description automatically generated

1. For the App Service Plan, complete the following steps:

* Under "Linux Plan," select "Create New" and then enter "project1plan."
* Under "Sku and size," select "Change size."
* The spec picker will pop up on the right-hand side of your screen.
  1. Note that this allows you to choose the pricing structure of your web app.
  2. Select "Dev/Test" and "Plan B1" (the green option), and then click "Apply," as the following image shows:



1. Leave the default options for all the other tabs. Select the "Review + Create" tab.
2. Select "Create" at the bottom of the screen to create your web app

Part 2a: Choose Domain with Godaddy

1. Navigate to GoDaddy.com
2. Pick a website for your cyber blog

rpwebapp.info

1. Complete your payment.

Part 2b: Map your custom Domain to Azure’s app services

1. Return to app that created in azure.
2. Select “Custom Domains” from the left-hand menu. Click “Add Custom Domain”. In “custom domain” text box, enter the new domain which you created and pay for on GoDaddy, and click validate.

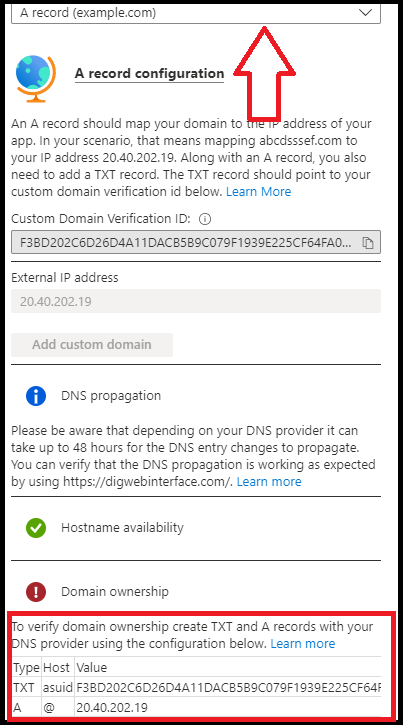
rpwebapp.info

Graphical user interface, text, application

Description automatically generated

1. After validating, the page will confirm whether the host name is available.

* Select the hostname record type "A Record," and notice the TXT and A data under "Domain ownership," as shown in the following image:



This will different all time. You will update this data in GoDaddy DNS record to prove that you own this domain.

1. Return to your GoDaddy.com products page: <https://account.godaddy.com/products>.
2. This page finds the domain that you just added. And select “DNS”.

Graphical user interface, application

Description automatically generated

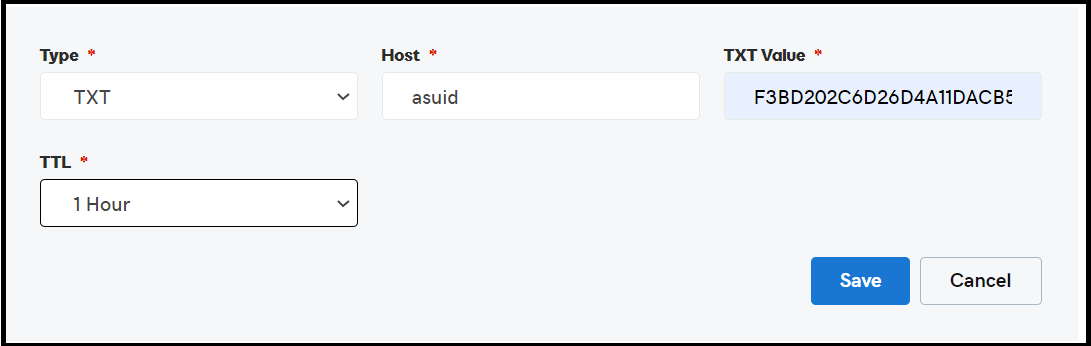
1. Below your existing DNS records, select “ADD” with out changing anything.

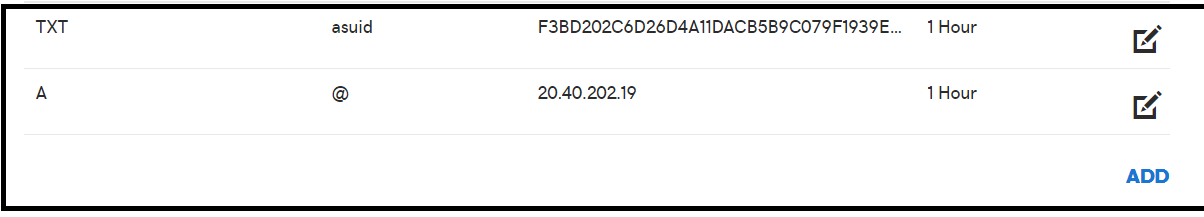
Graphical user interface, text, application, email

Description automatically generated

1. One at a time, add in the two “Domain Ownership” records (TXT and A), that you can see in image of point num 6, and “SAVE” after each one.

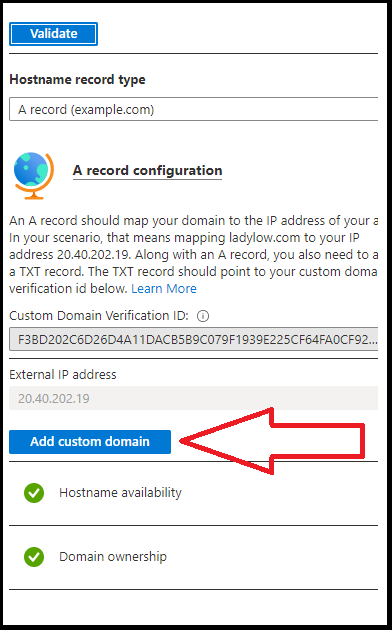
The TTL can stay at 1 hour.





1. Return to Azure and repeat the step 5. After clicking “Validate”, green check marks should appear next to “Hostname Availability” and “Domain ownership”.

1. Select “Add Custom Domain”.



1. After a minute, your new domain will appear under “Assigned Custom Domain”.

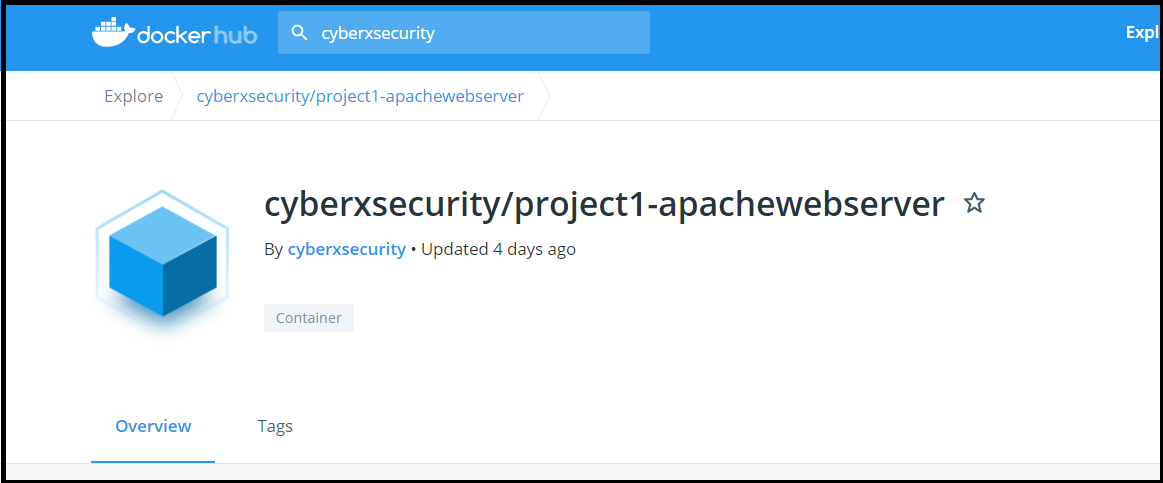


Graphical user interface, text, application, email

Description automatically generated

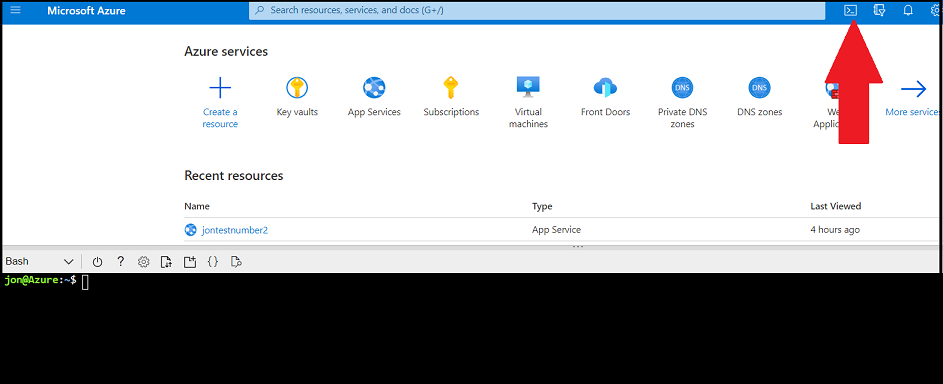
Part 3: Deploy a Container on Web App

1. For your web application, you will use a Docker container that has been added to Docker Hub. View the Docker container at the following location: [Cyber Blog Framework - Docker Container](https://hub.docker.com/r/cyberxsecurity/project1-apachewebserver).
2. Note that the Docker container image name is cyberxsecurity/project1-apachewebserver, as the following image shows:



1. Next, you will use the Azure Cloud Shell to deploy this container to your web application.

* Azure Cloud Shell takes user input from a command line to manage Azure's cloud resources.
  + While we will use Bash, you can also use Powershell to administer your commands.
  + For additional resources on Azure's Cloud Shell, refer to the following pages:
    - [Azure Cloud Shell Overview](https://docs.microsoft.com/en-us/azure/cloud-shell/overview)
    - [Azure Web App Container commands](https://docs.microsoft.com/en-us/cli/azure/webapp/config/container?view=azure-cli-latest)
* To open Azure Cloud Shell, click the shell logo in the tool bar at the top of the screen, as indicated by the red arrow in the following image:



* Once you've clicked this icon, the Cloud Shell will be accessible at the bottom of your page.
* When using Shell, you may receive the following prompts:
  + Select which shell to use (Bash or PowerShell): Select "Bash."
  + Create Storage: If a window appears, select "Create Storage."

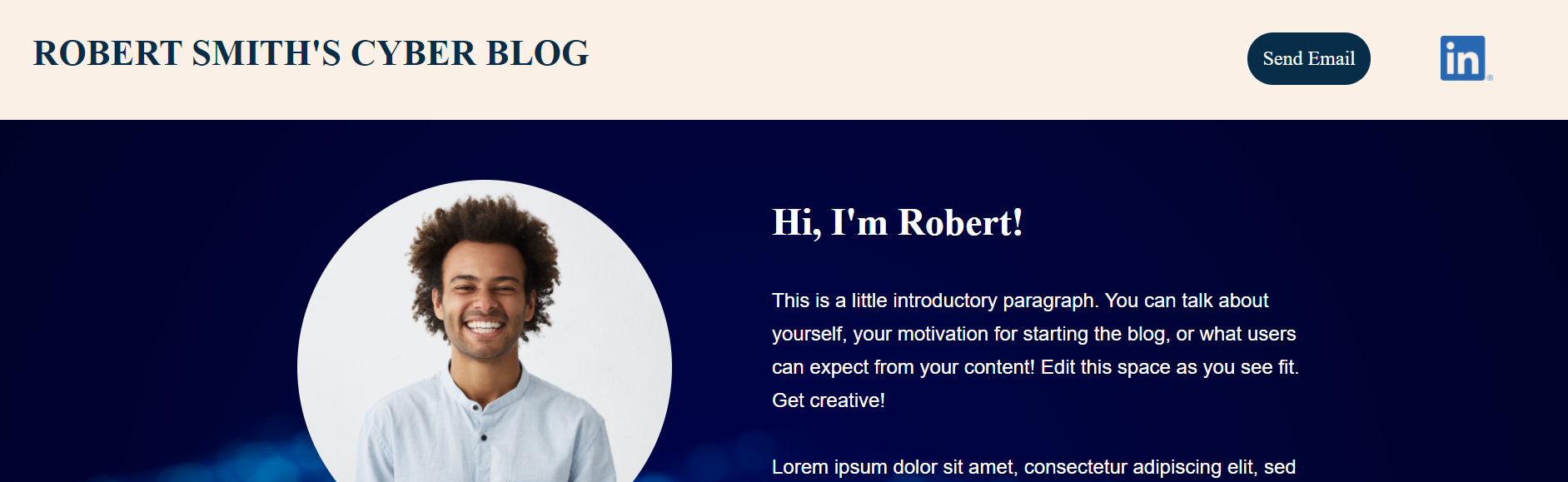
1. From the command line you will enter a command to configure your container.
   * There are three types of commands that manage your web app container settings:
   * (1) az webapp config container delete - This will delete your web app container's settings.
   * (2) az webapp config container set - This will set your web app container's settings.
   * (3) az webapp config container show - This will display the current details of your web app container's settings.
2. Run the following commands to configure container and 2nd command is to show that container has been added correctly.

Text

Description automatically generated

1. Now, check the unique domain that you selected to verify that the container has been successfully deployed.

* A cyber blog webpage that looks like the following image should appear (note that it may take five to eight minutes to load):



Now you are ready to customize your website.

Part 4: Design your custom Web App

The container that you just loaded onto your web application is a framework for a cyber-blog page that you can customize.

You will now customize the following elements of the webpage:

* Your name
* Your email
* Your LinkedIn profile link
* Your introduction
* Your picture
* Two custom blog posts on topics of your choice

1. To design and customize your webpage, you'll need to access the HTML pages of your new web application.
   * To access these pages, you need to SSH over to your container and access the HTML files.
   * Return to your web app in Azure, select "SSH" from the left-hand toolbar, and then select "GO," as shown in the following image:

Graphical user interface, application, Word

Description automatically generated

1. This will SSH you right into the container.

* Once you have access, change directories to the location where the HTML files are located by running cd /var/www/html, as the following image shows:

Text

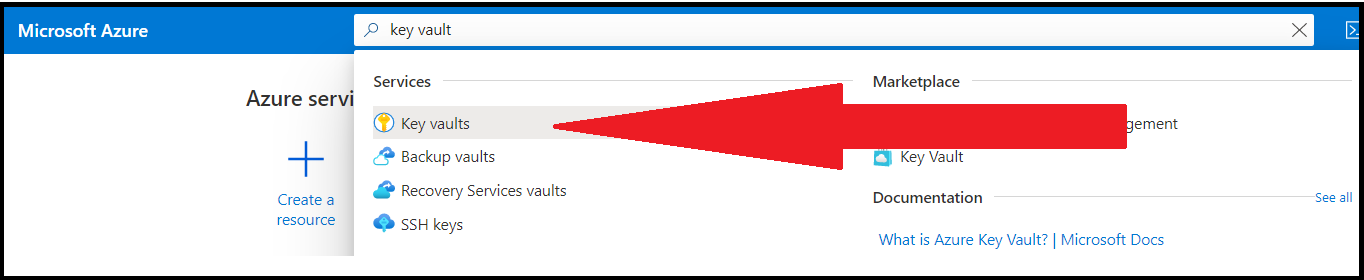
Description automatically generated

1. Run nano index.html to customize html file. For html file go to project 1 folder-day 1-html script

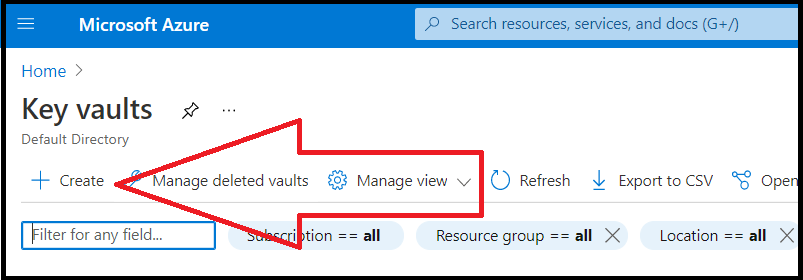
Day 2: Secure web app with SSL Certificate

Part 1: Create a Key Vault

* 1. Select "Key vaults" from the Azure search field at the top of the page, as the following image shows



1. Select "+ Create" from the Key Vault page to create your key vault.



1. “Create key vault”

* Subscription/Resource Group: CloudWebApp
* Key Vault Name: RP-Proj1
* Region: Canada Central
* Pricing tier: Select the "Standard" tier.
* Leave the default options for all the other tabs (Access Policy, Networking, Tags).

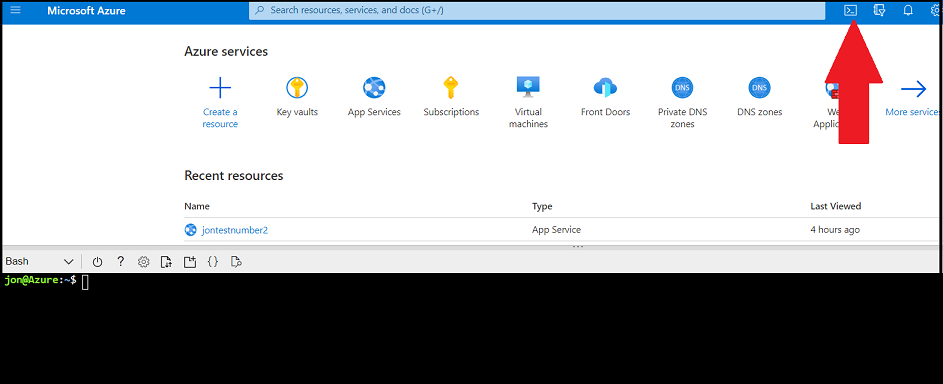
Graphical user interface, text, application, email

Description automatically generated

1. “Review + Create”

Part 2: Create self-signed Certificate

* 1. Access the same Cloud Shell like day 1 to load the docker container.



* 1. Run the commands as per Screen Shot.

In the form, Common Name: rpwebapp.info

Text

Description automatically generated

You can see your newly created .key and .crt by running ls in the image. Note that Azure requires a PFX format for its certificates.

The PFX format is the server certificate and the private key combined into a single encrypted file. To create a PFX format, run the next command from the image.

Now enter Password twice as per it prompts. And do not forget as you will need that again in short time.

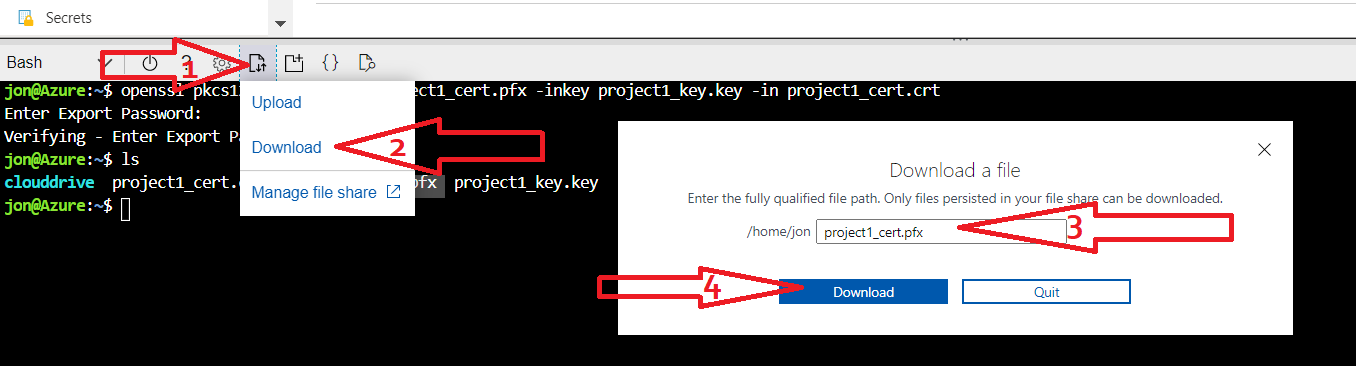
* 1. To download your new PFX certificate, complete the following four steps:

(1) Click the "Upload/Download" icon in the toolbar above your Cloud Shell window.

(2) Select "Download."

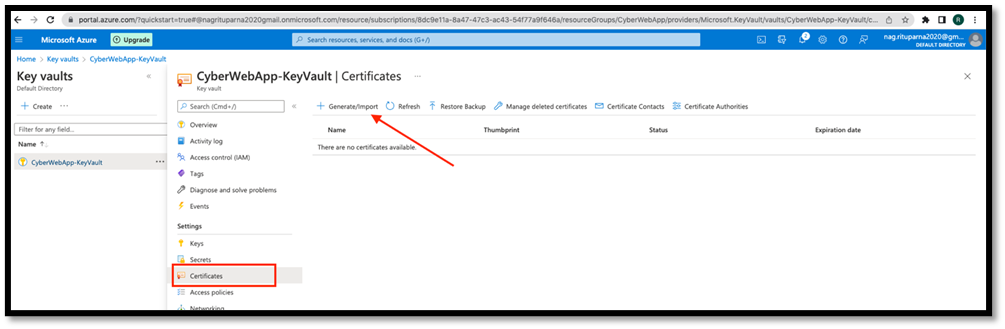
(3) Enter the name of your PFX certificate in the "Download a file" window.

(4) Click "Download."

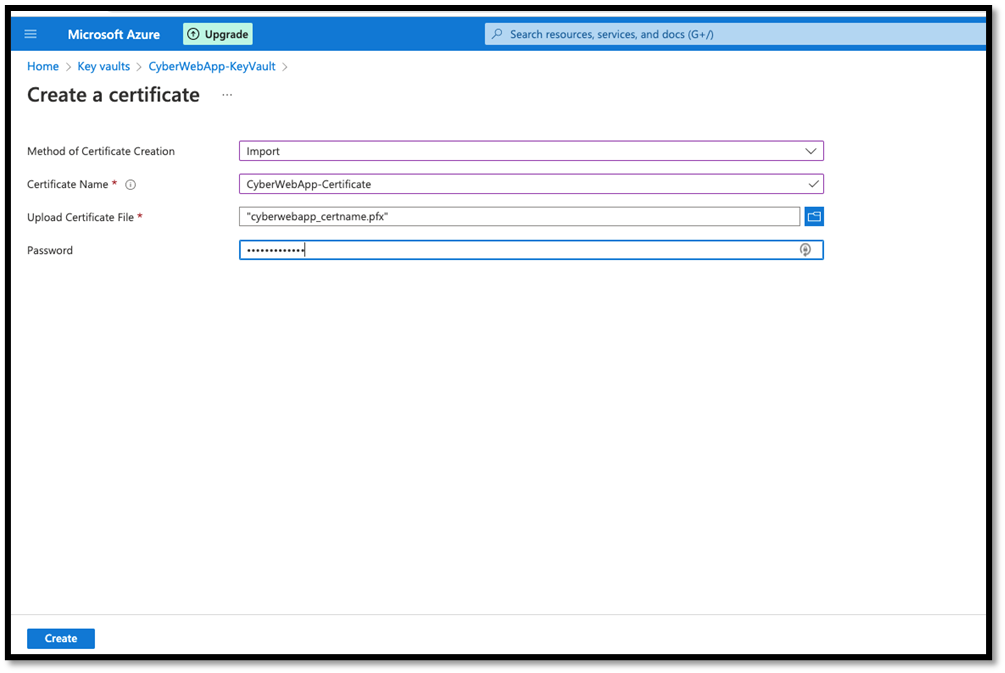


Part 3: Import and Bind your Self-signed Certificate to your web app

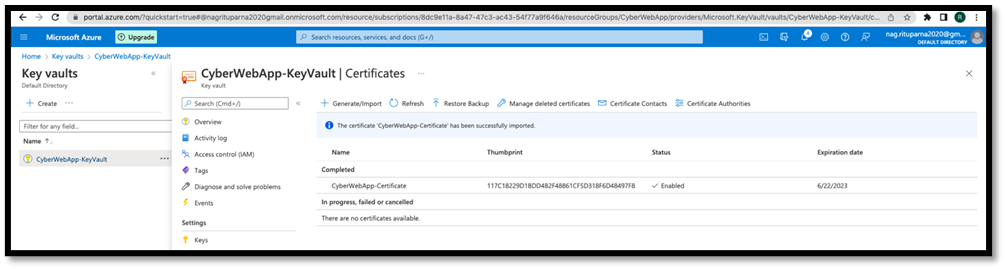
* 1. Go to Azure and create a certificate



* 1. On the "Create a certificate" page, select the following:
* Method of Certificate Creation: Import
* Certificate Name: proj-1
* Upload Certificate File: Select your PFX certificate (it's likely in your Downloads folder)
* Password: Enter the password that you created in Part 2



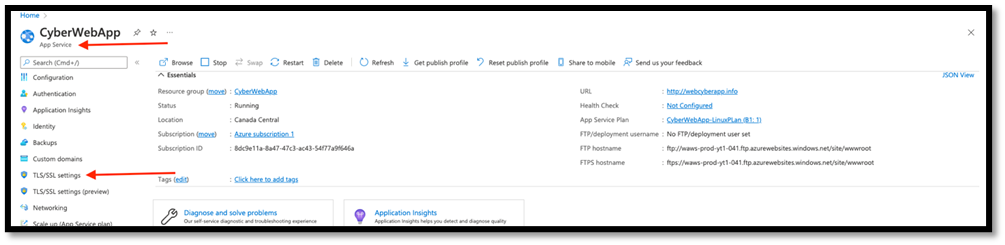
* 1. Select "Create" to upload your certificate.
* The following success message should appear to confirm that your PFX certificate has been uploaded to your key vault



* 1. Now that you have uploaded your certificate, it's time to add it to your web application. To do so, complete the following steps:

Return to the web application (under "App Services") that you created on Day 1.

On this page, select "TLS/SSL Settings," as the following image shows:

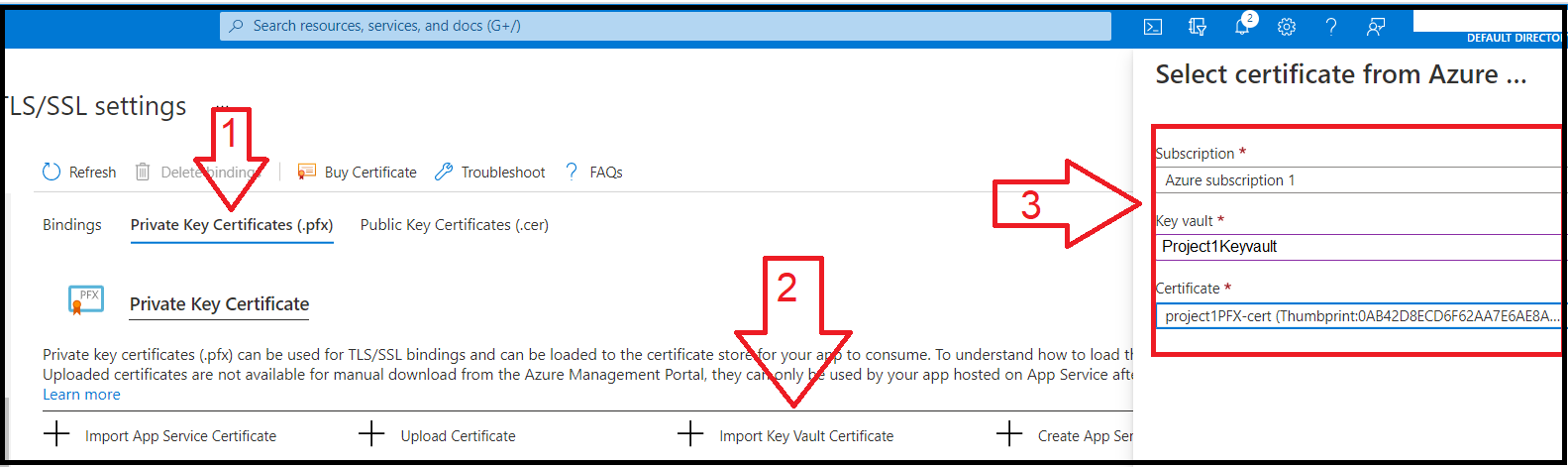


5. On this page, import your new PFX certificate from your key vault. To do so, complete the following steps:

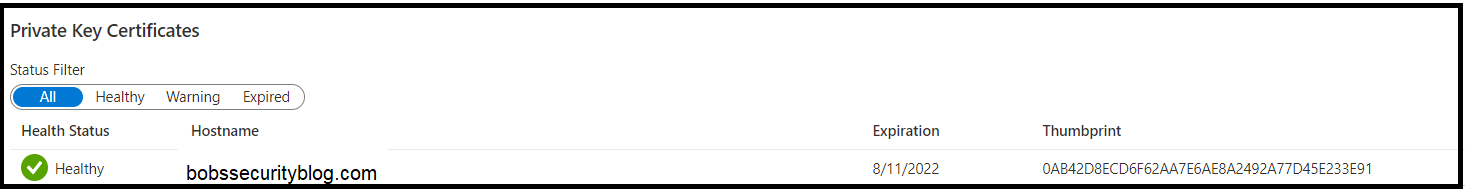
(1) Select "Private Key Certificates."

(2) Click "+ Import Key Vault Certificate."

(3) When the pane opens on the right side of your screen, enter your subscription, key vault, and certificate that you just created.

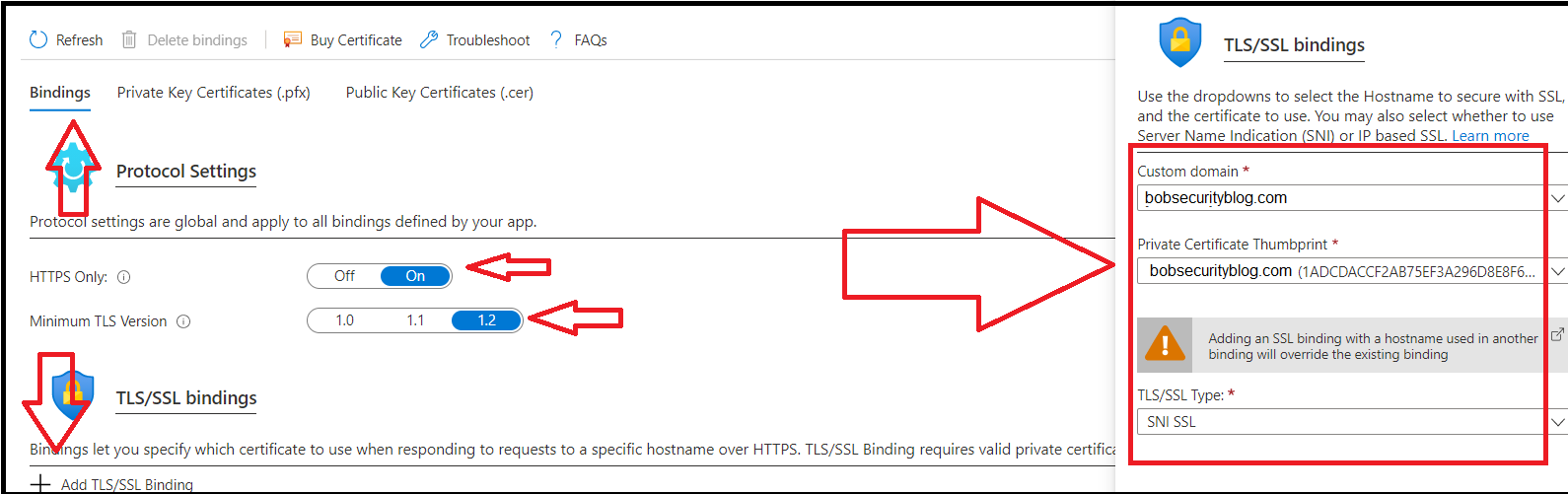


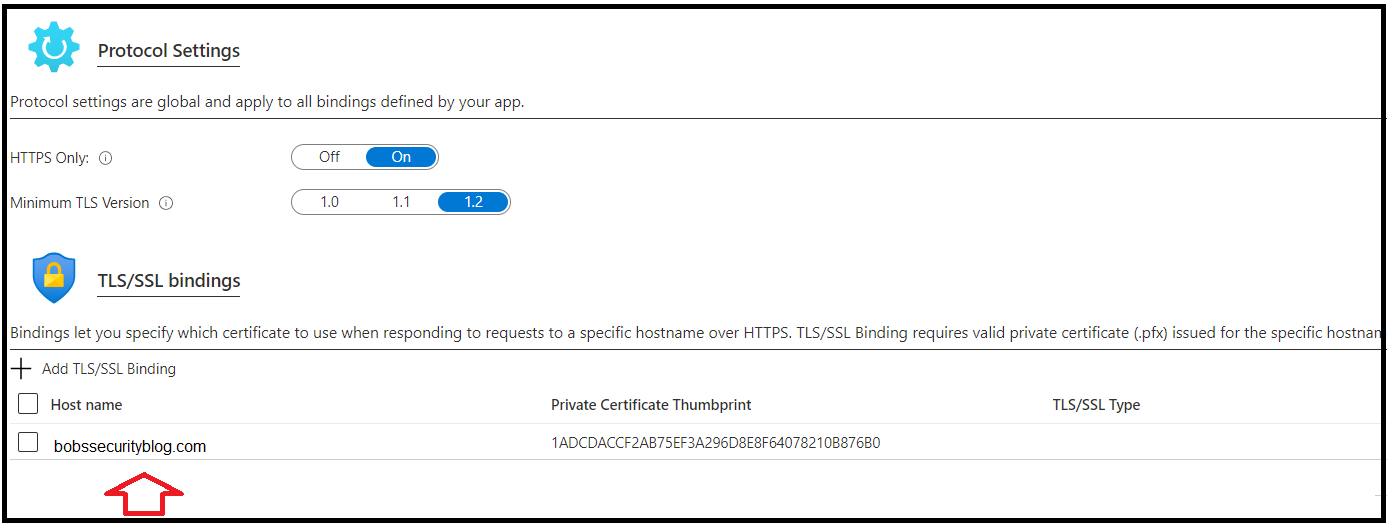
1. Your certificate should now appear as an available certificate to use for your web application



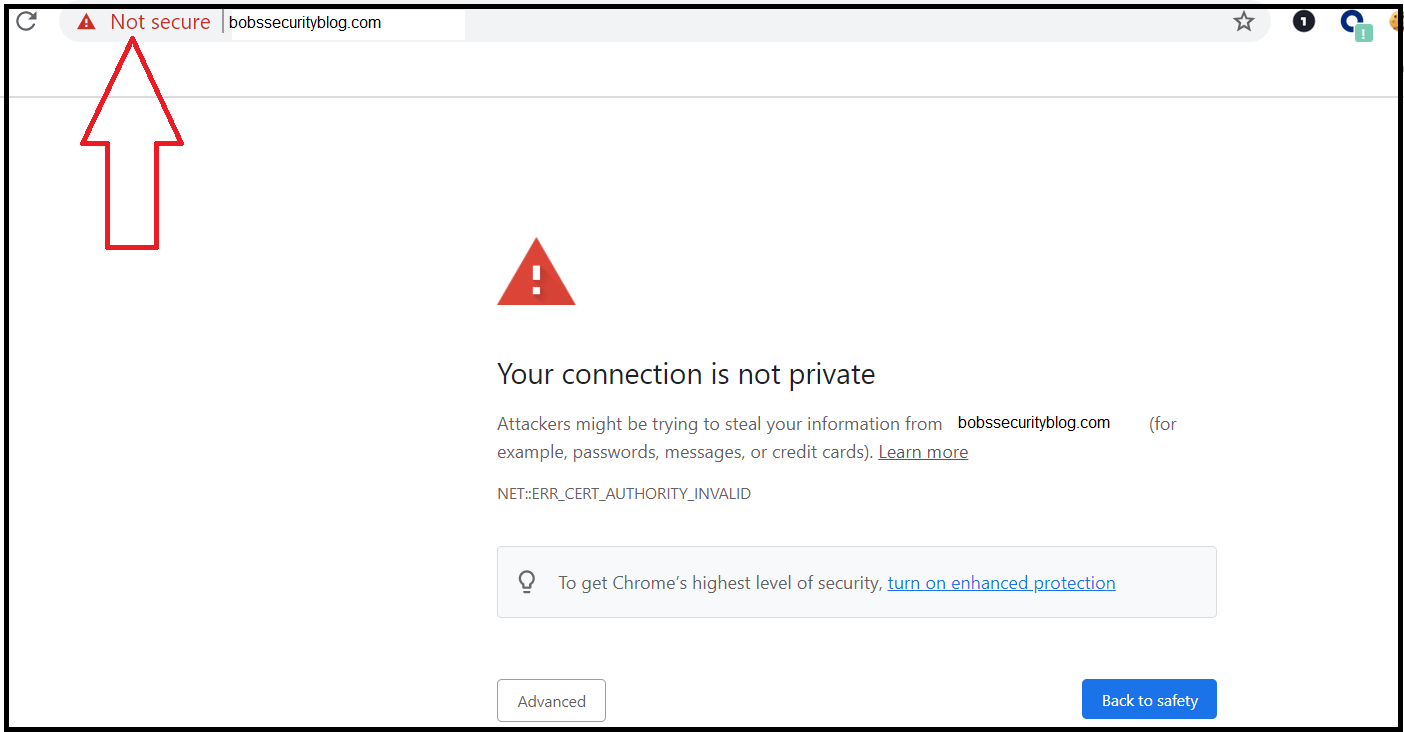
1. Currently your certificate is just available to use for your web application—now, it is time to bind it! To bind your certificate, complete the following steps:

* Select "Bindings."
* Toggle "HTTPS Only" to "On."
* Toggle "Minimum TLS Version" to "1.2."
* Click "+ Add TLS/SSL Binding."
* When the pane appears on the right side of your screen, select your domain, the matching certificate, and "SNI/SSL" for the TLS/SSL type.

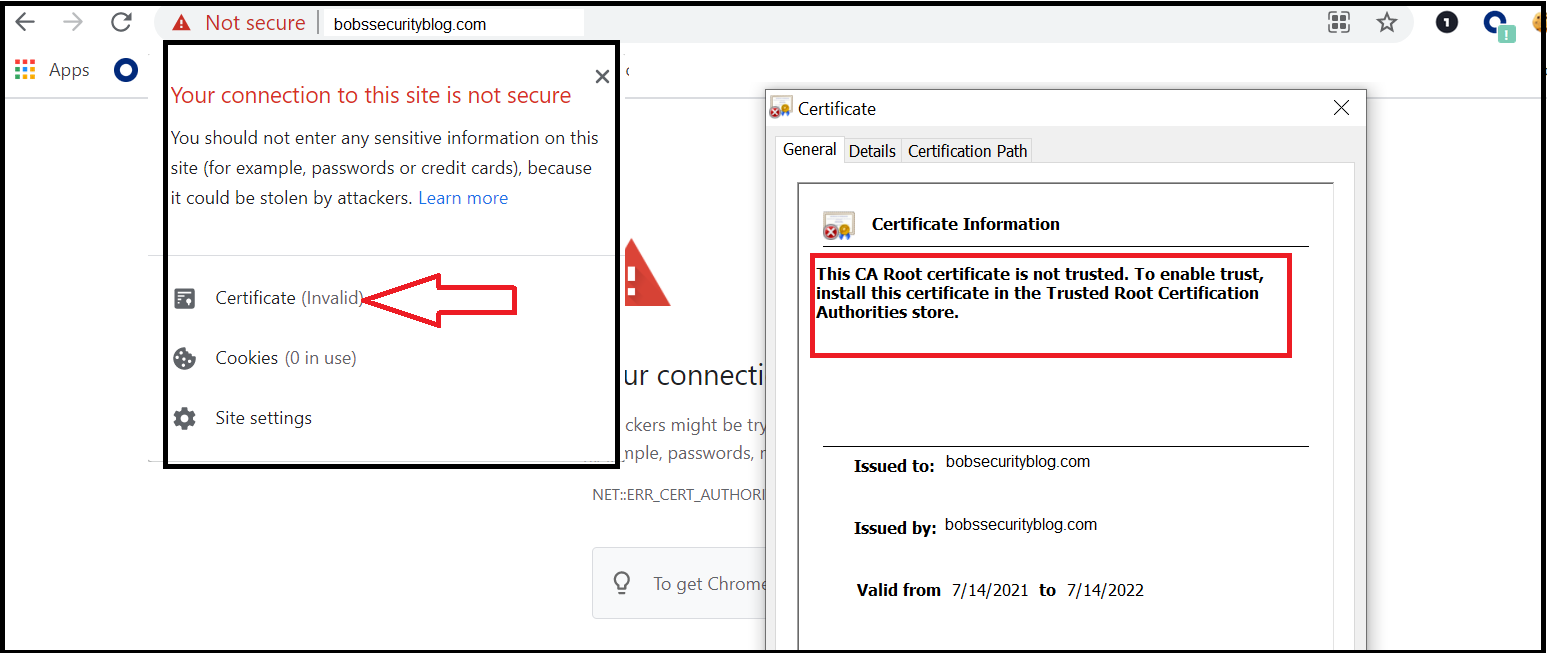




1. Now, open a browser and access your web application. Did your browser return an error like the one shown in the following image?

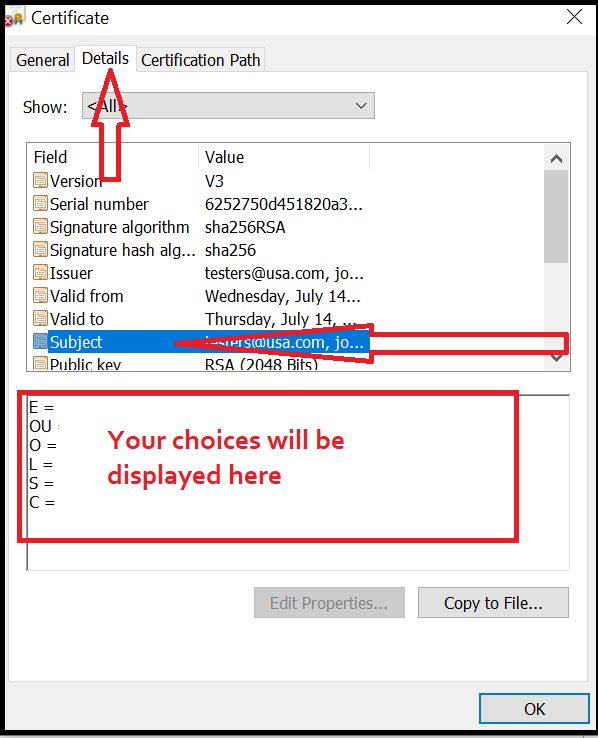


1. Let's examine the certificate that you just added. Click "Not secure" in the search bar if you are in Chrome, or a similar message depending on your browser



* + After selecting "Not secure," select "Certificate (Invalid)" from the menu to examine your certificate.
  + Note the reason for your error based on the message on your certificate. This message is due to the fact that your certificate was created by you and not a trusted CA.

1. Next, click the "Details" tab of your certificate, then select the "Subject" option.

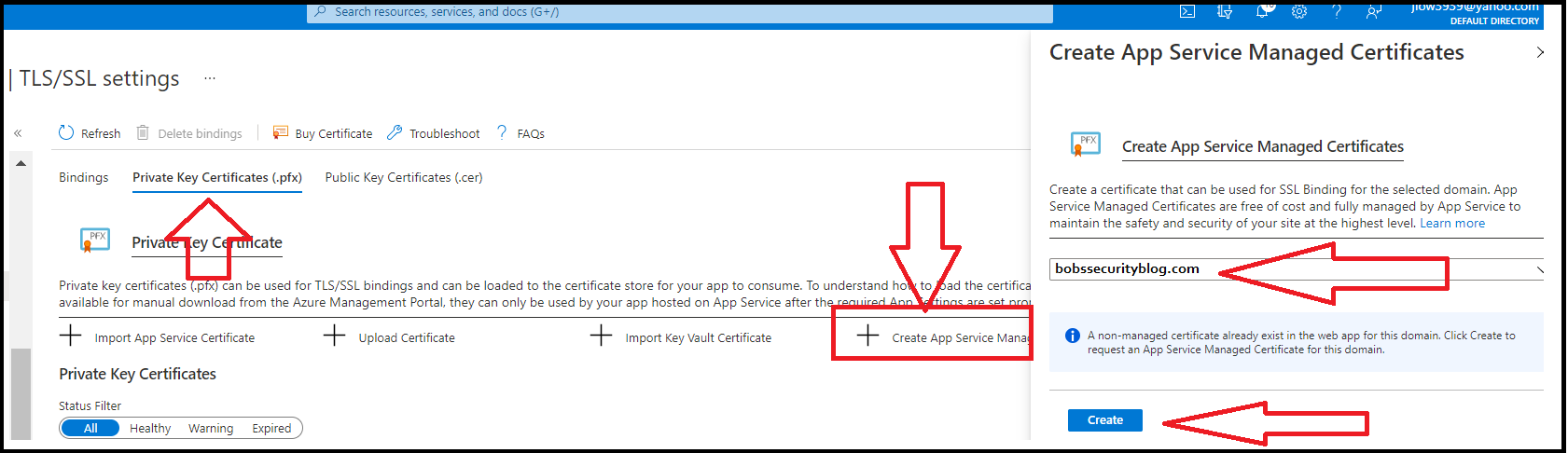


* + Note the results that now display in the box on the bottom; these were the options that you selected when you created your certificate with OpenSSL.

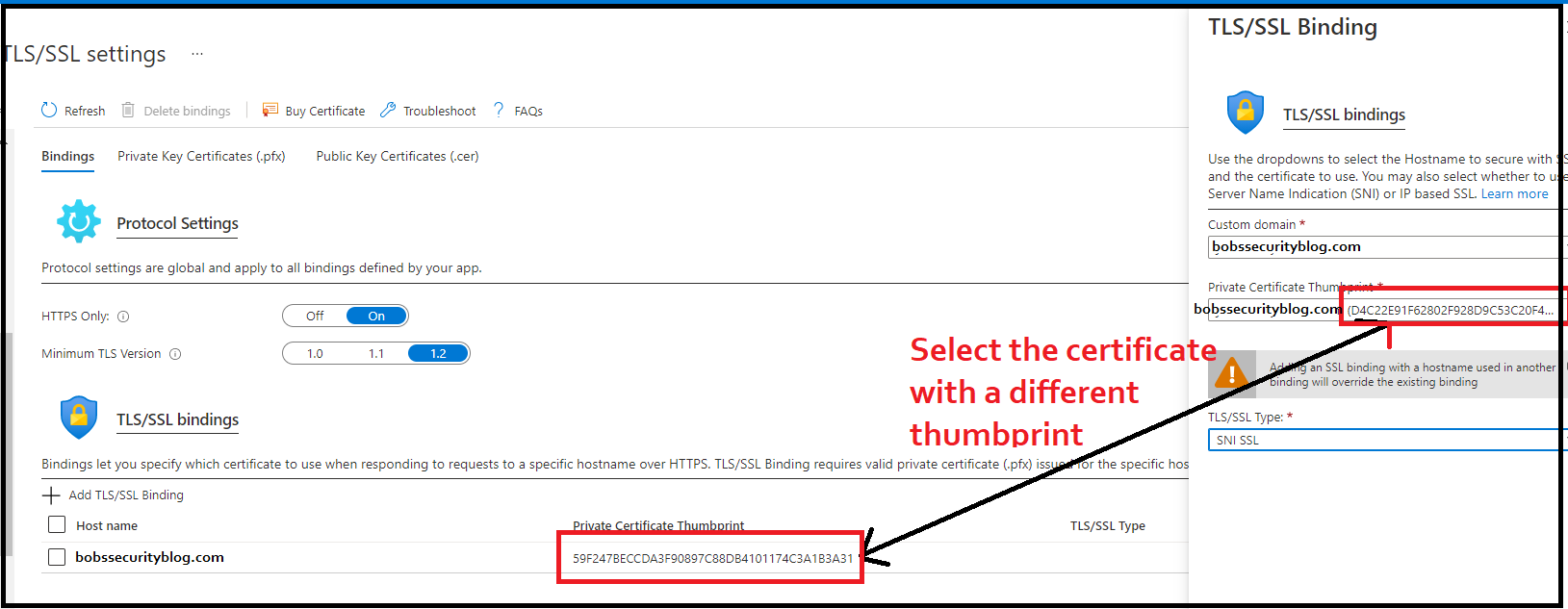
You have successfully created a self-signed certificate and bound it to your web application using Azure!

Part 4: Create and Bind an App Service Managed Certificate

1. First, return to "TLS/SSL settings" under your web application.
2. Select "Private Key Certificates."
3. Select "+ Create App Service Managed Certificates."
4. When the pop-up appears on the right side of your screen, select your domain and click "Create".



1. Once your app service managed certificate has been created, return to the "Bindings" tab.
2. Select "+ Add TLS/SSL Binding."
3. When the pane appears on the right side of your screen, select your domain and the new certificate that you just created.
   1. Note the current thumbprint next to your domain under the "TLS/SSL bindings" menu and select the certificate with the other thumbprint.



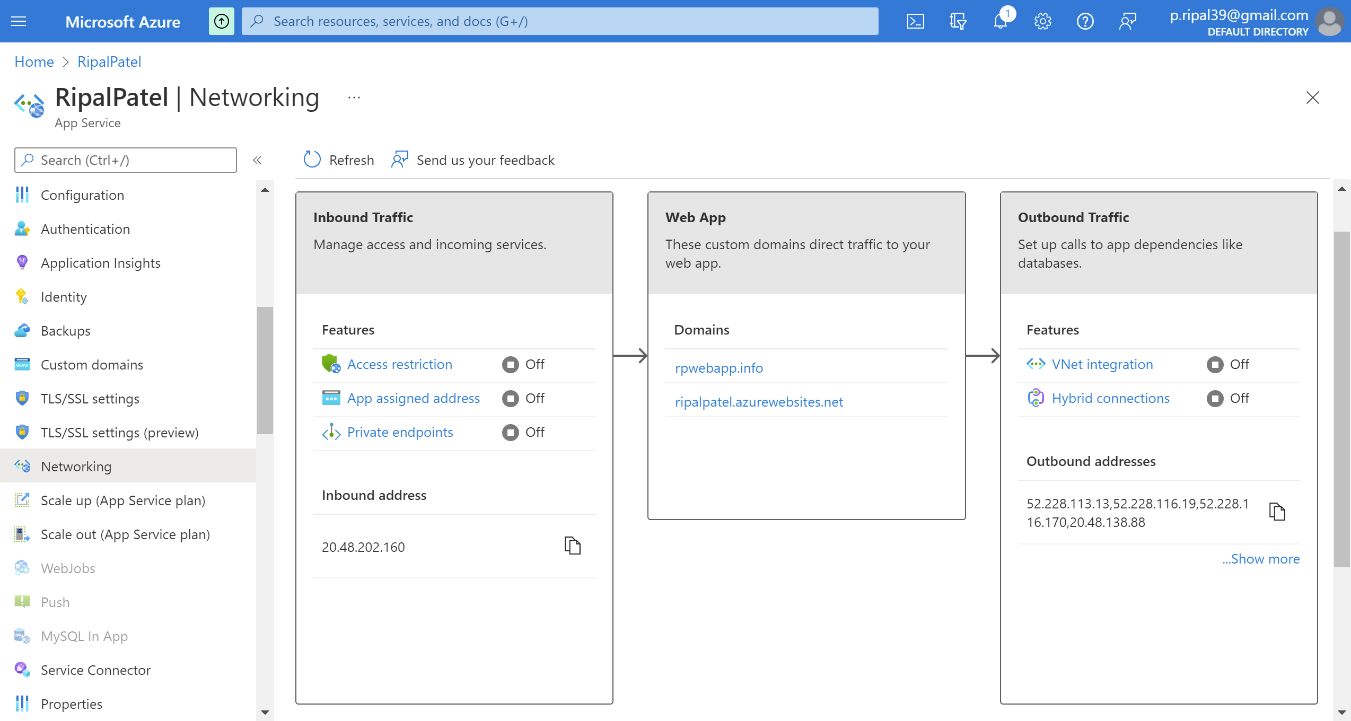
1. Select "SNI/SSL" for the TLS/SSL type, then click "Add Binding."
2. Now that your new app services managed certificate has been bound to your web application, revisit your website. You should not see any warnings displayed this time!

Congratulations, you have now created a web application and secured it with a trusted SSL certificate!

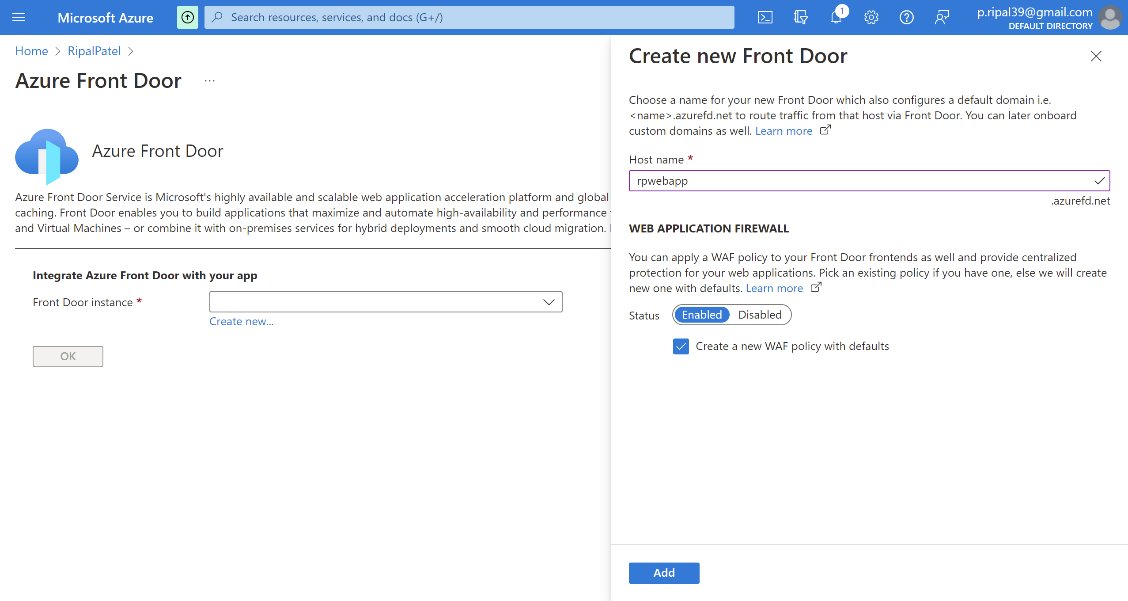
Day 3: Protecting Web Application with Azure’s Security Features.

Part 1: Create Front Door Instance

1. Access the app service resource that you created on Day 1.
2. From the menu on the left side of the screen, select "Networking."
3. From this page, select "Azure Front Door" under "More networking features".

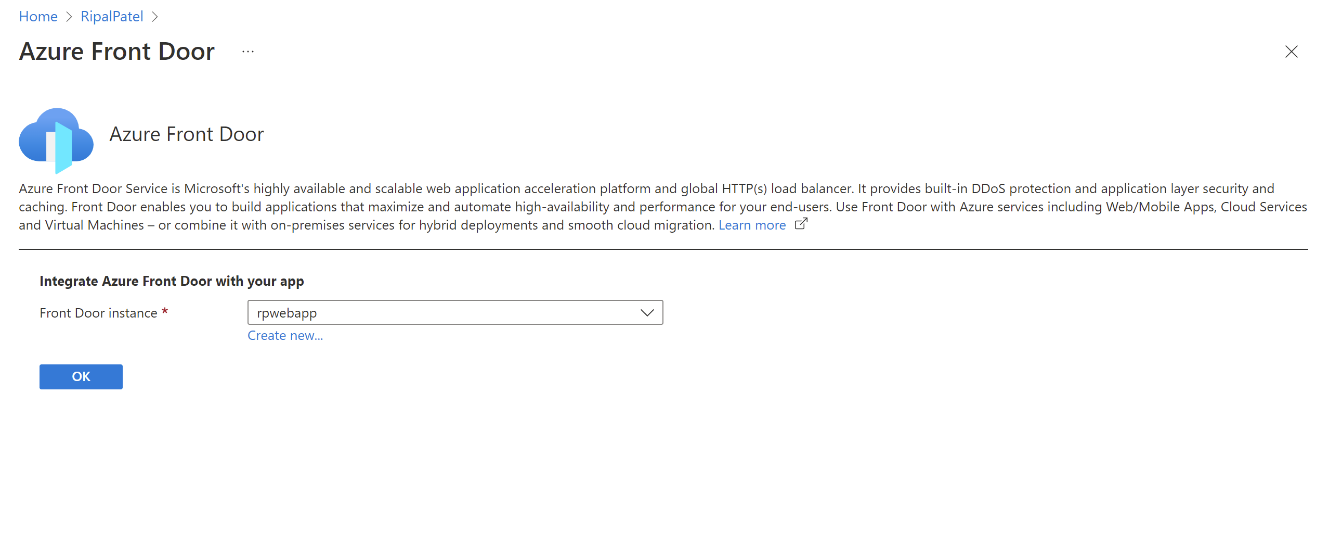


1. On the next page, since you haven't created your Front Door resource yet, select "Create new" under "Front Door instance."
2. This will open a pane on the right side of your screen.
   1. In this pane, name your Front Door "project1-FrontDoor".
   2. Leave the default settings to create a default **web application firewall (WAF)**.
   3. Click the "Add" button at the bottom of the pane

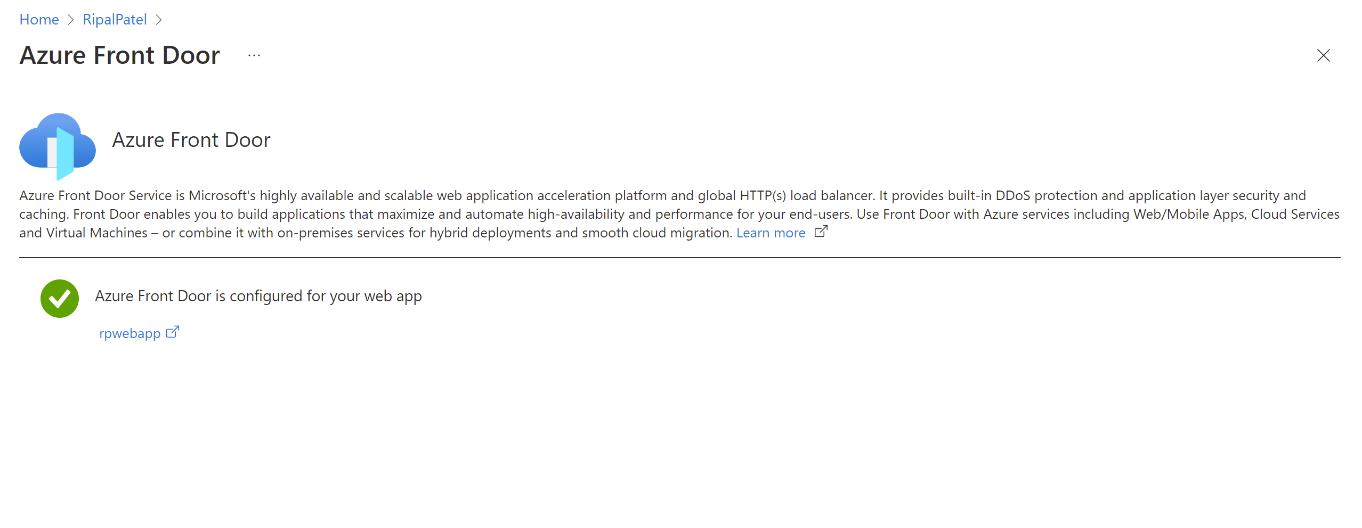


1. This will return you to the Azure Front Door page.

* Click "OK" to update the Front Door instance to your application

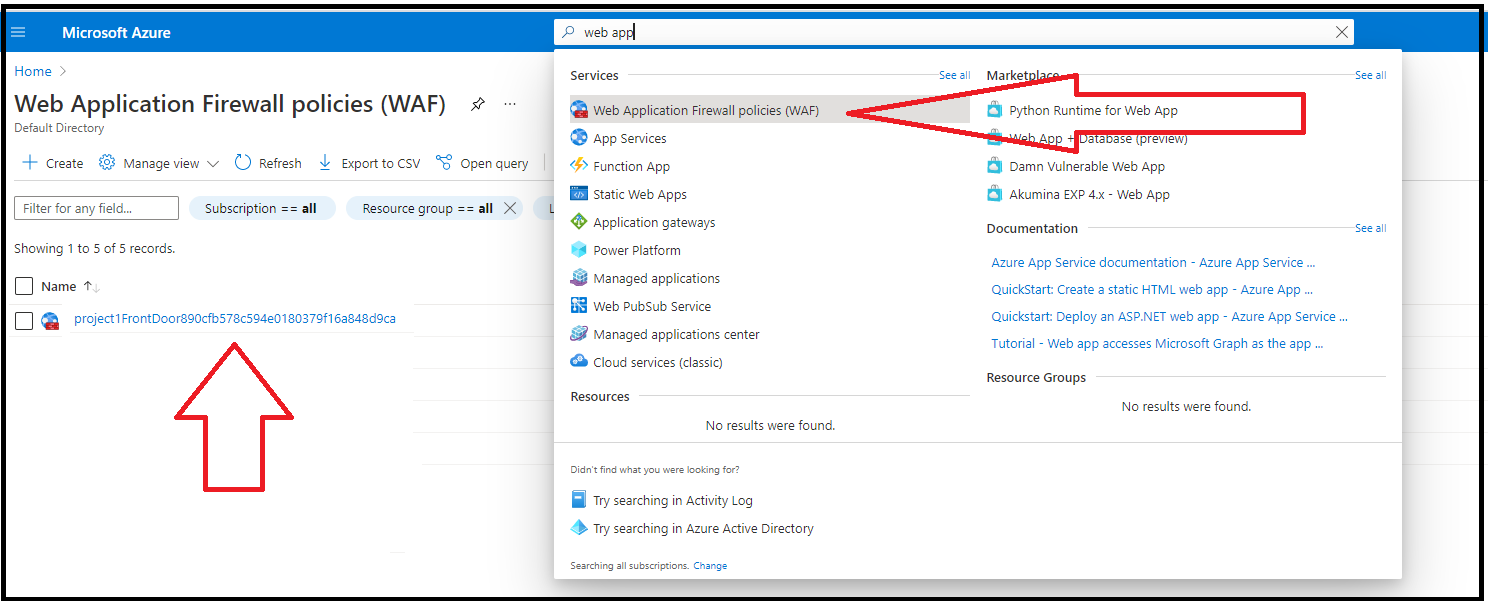


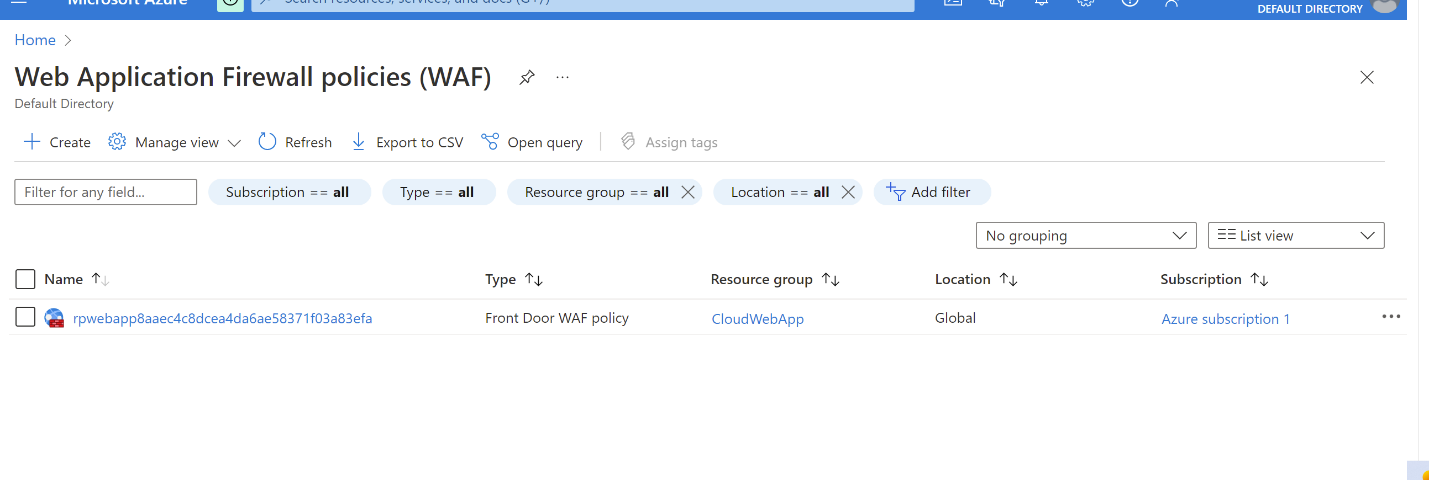
1. To verify that your Front Door instance has been set up correctly, select "Azure Front Door" (from Step 4) again.
2. The message "Azure Front Door is configured for your web app" should display as confirmation.



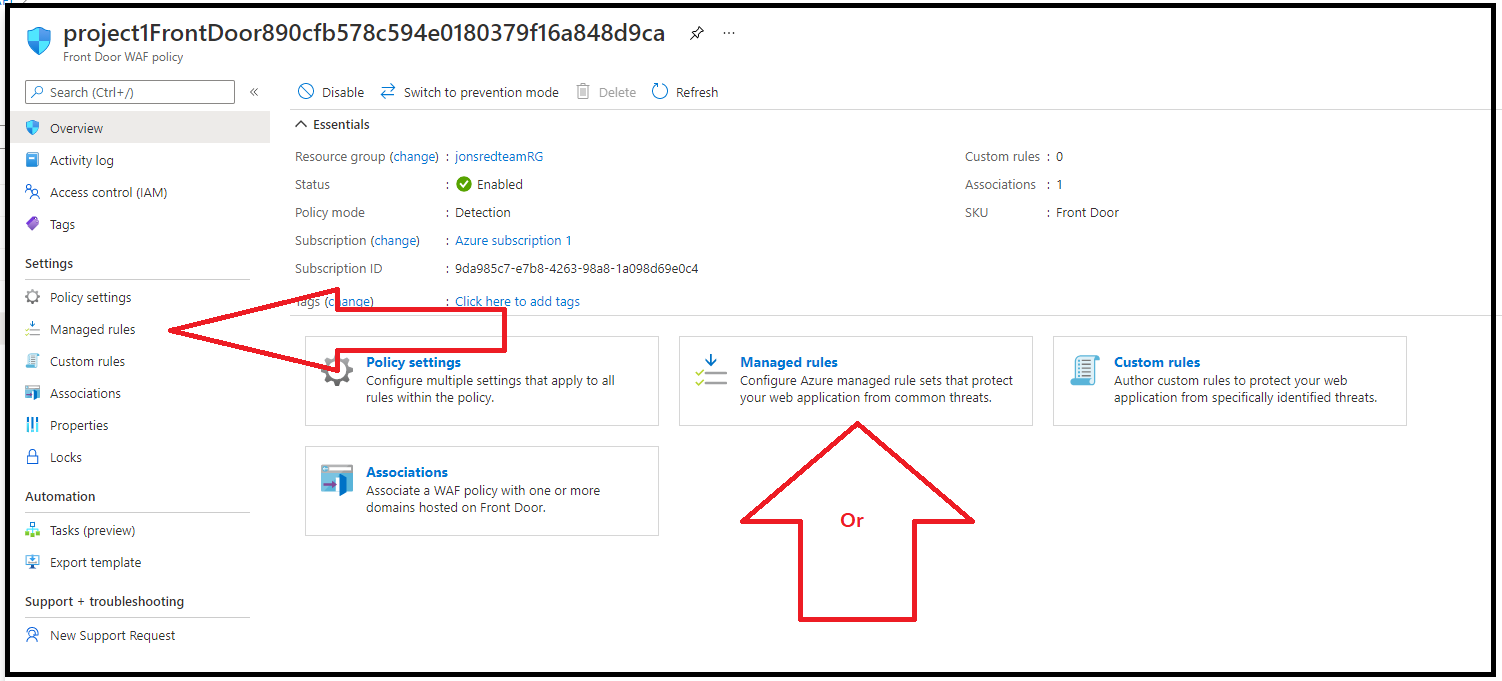
Part 2: Analyze WAF Rule Sets

1. From your Azure portal, enter "web app" until "Web Application Firewall policies (WAF)" appears as one of the choices in the dropdown.
2. Select that option. The WAF that you created during the previous step should display on the "Web Application Firewall policies (WAF)" page.
   * Note: It will begin with "project1frontdoor" and end with several random letters and numbers.
3. Select your WAF.

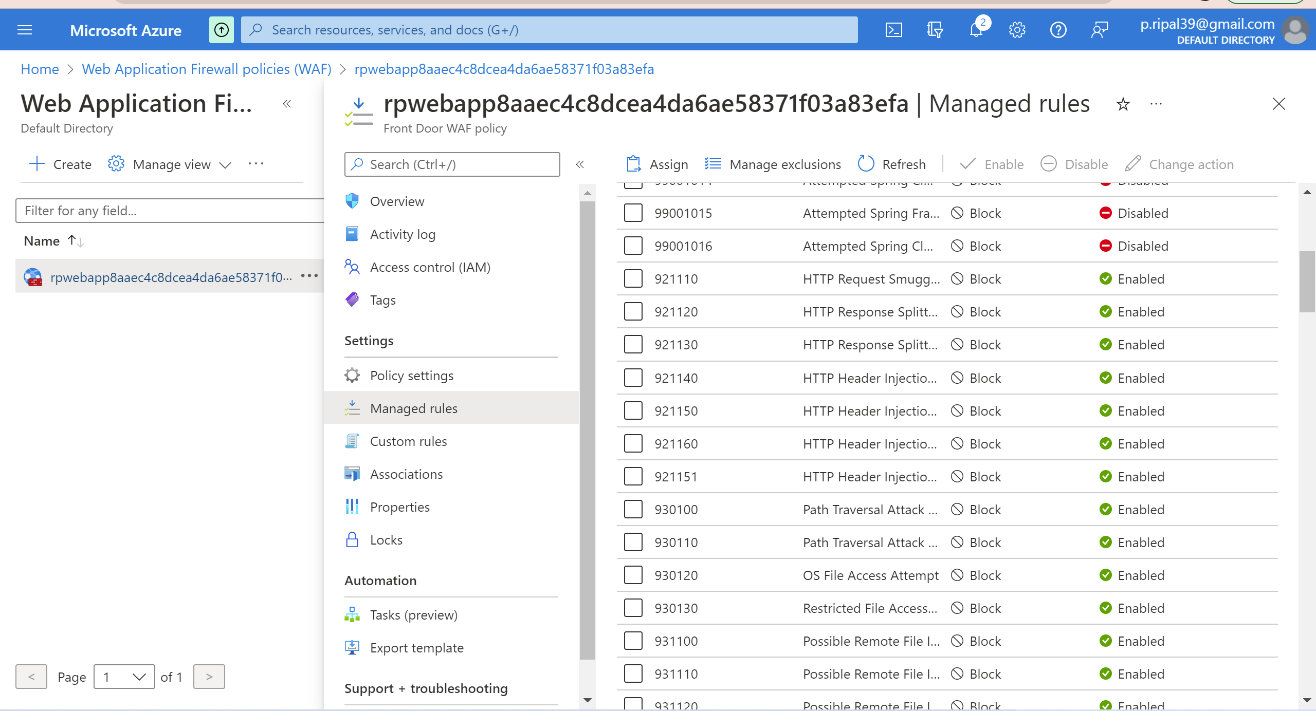




1. When your WAF policies page opens, notice the options on the left side of your screen.
2. Select "Managed rules" either from the left-hand toolbar or from the box on the bottom of the page.



1. When the "Managed rules" page appears, scroll through the page to view the various rules.

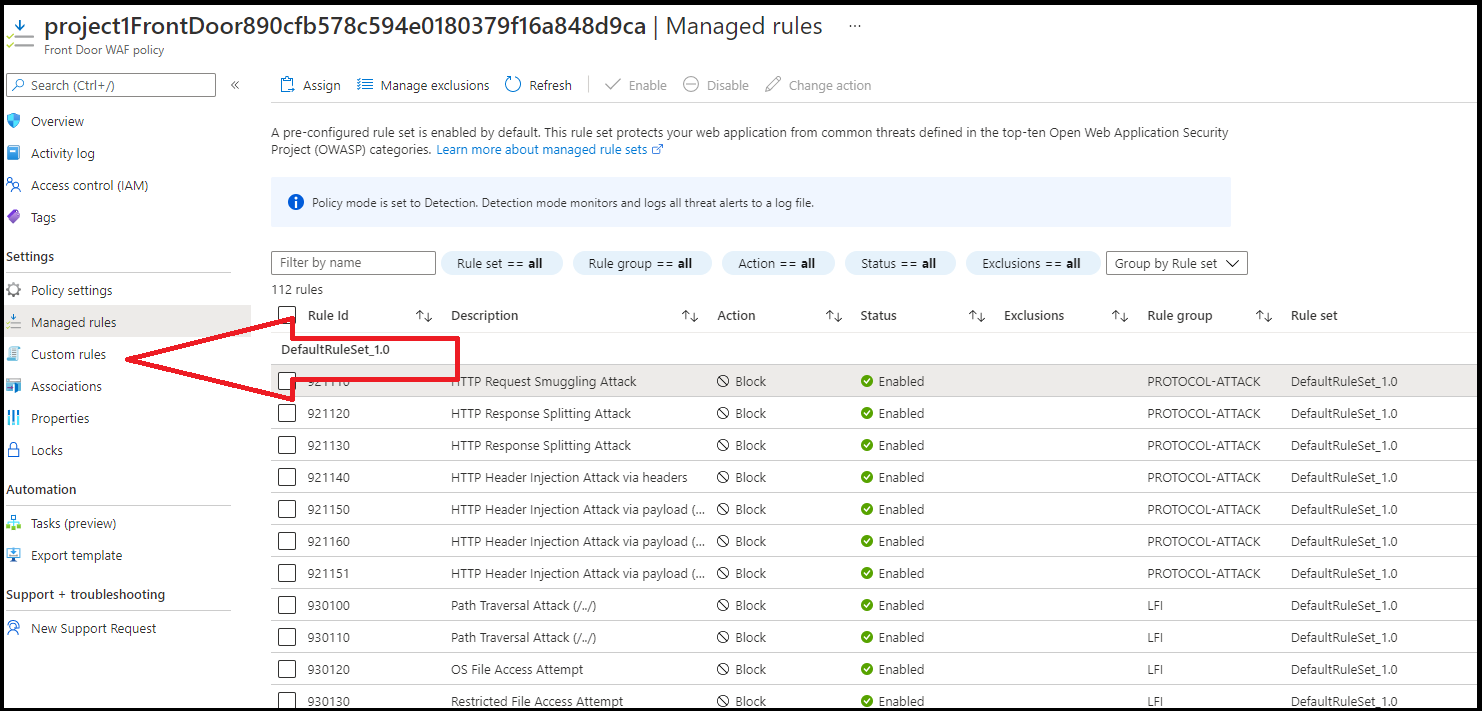


* Note the following about these rules:
  + This is the list of the application vulnerabilities that the WAF will protect against (we will explore these vulnerabilities in further detail in the Web Vulnerabilities unit).
  + While it's unlikely that your web application would be impacted by these vulnerabilities, this exercise illustrates the Azure WAF feature, which identifies and blocks the application attacks indicated on this page.
  + These managed rules can be individually enabled or disabled, and a variety of actions can be taken if an attack is identified, such as:
    - Allow the request.
    - Block the request.
    - Log the request.
    - Redirect the request to another webpage.

Part 3: Configure Custom WAF Rules

Configure a custom WAF rule to protect against a potential security attack. Following steps are for how to create a custom rule on web application to protect against variety of attacks from international IP addresses, and accept traffic from the US, Canada, and Australia locations where business partners reside.

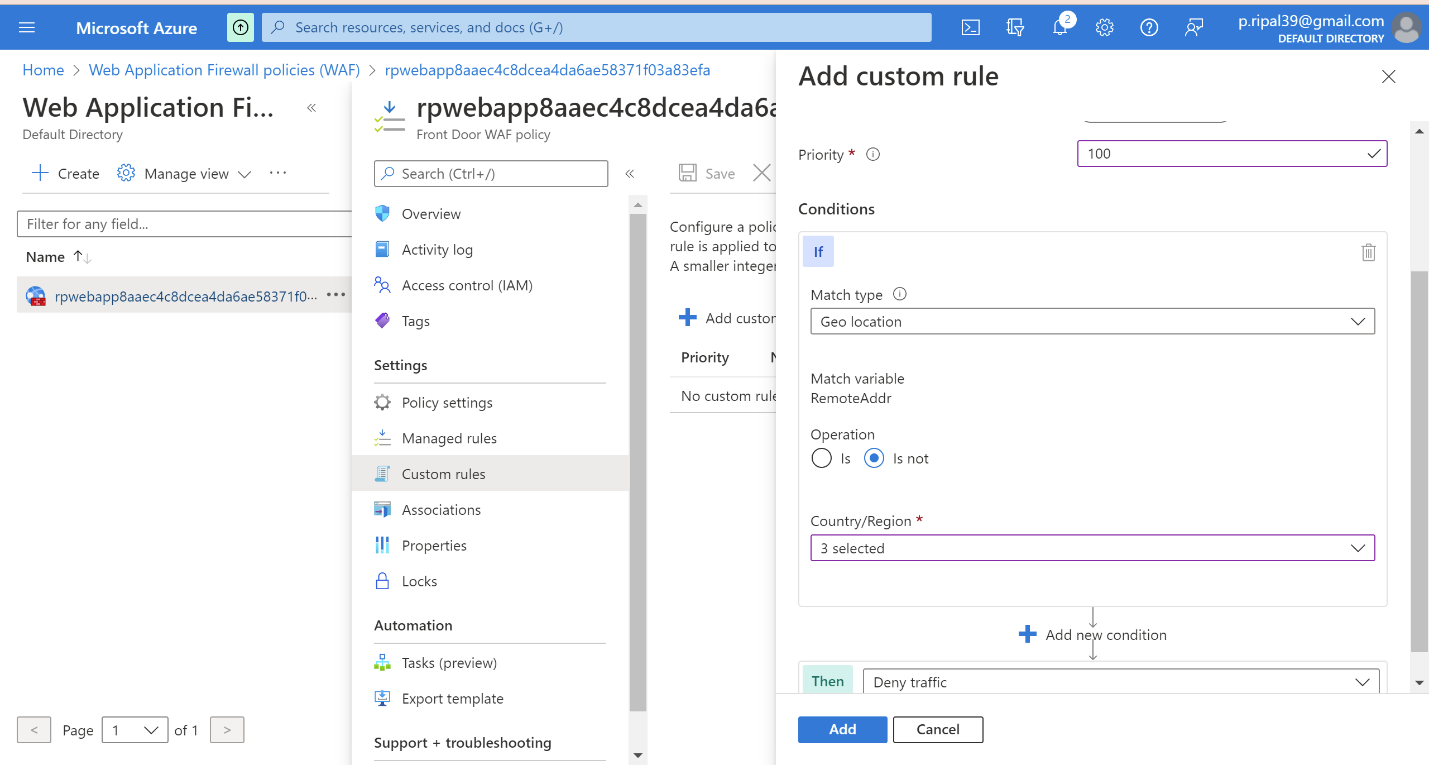
* 1. Select "Custom rules" from the toolbar on the left-hand side of the screen.



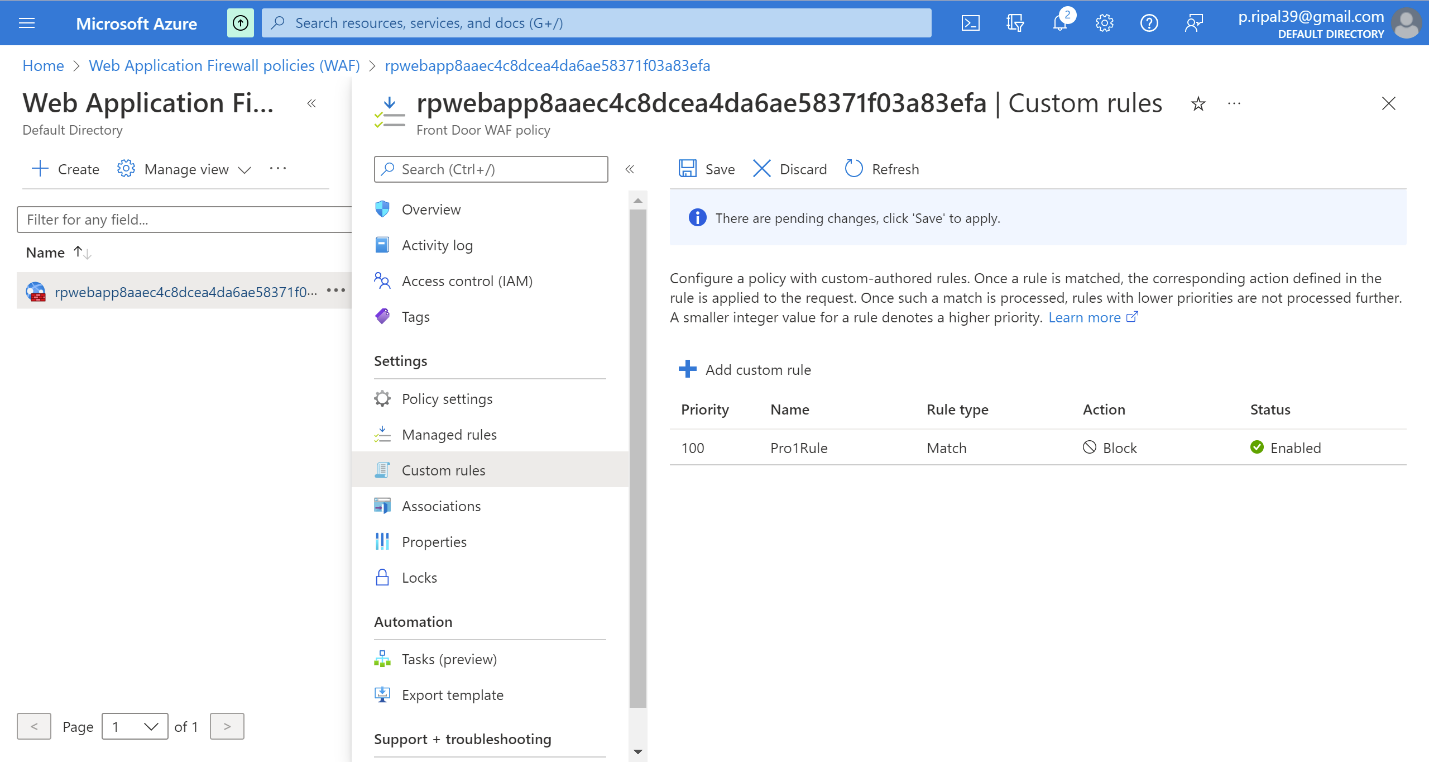
* 1. To create a custom rule, select "+ Add custom rule."
* When the pane pops up on the right, name your custom rule "Project1rule."
* Leave the status and rule type at the default options.
* Set the priority to 100.
* Set the following terms for the rule's condition:
* Match type: Geo location
* Operation: is not
* Select the three countries (USA, Canada, Australia)
* Then: Deny traffic
* Then, click "Add."

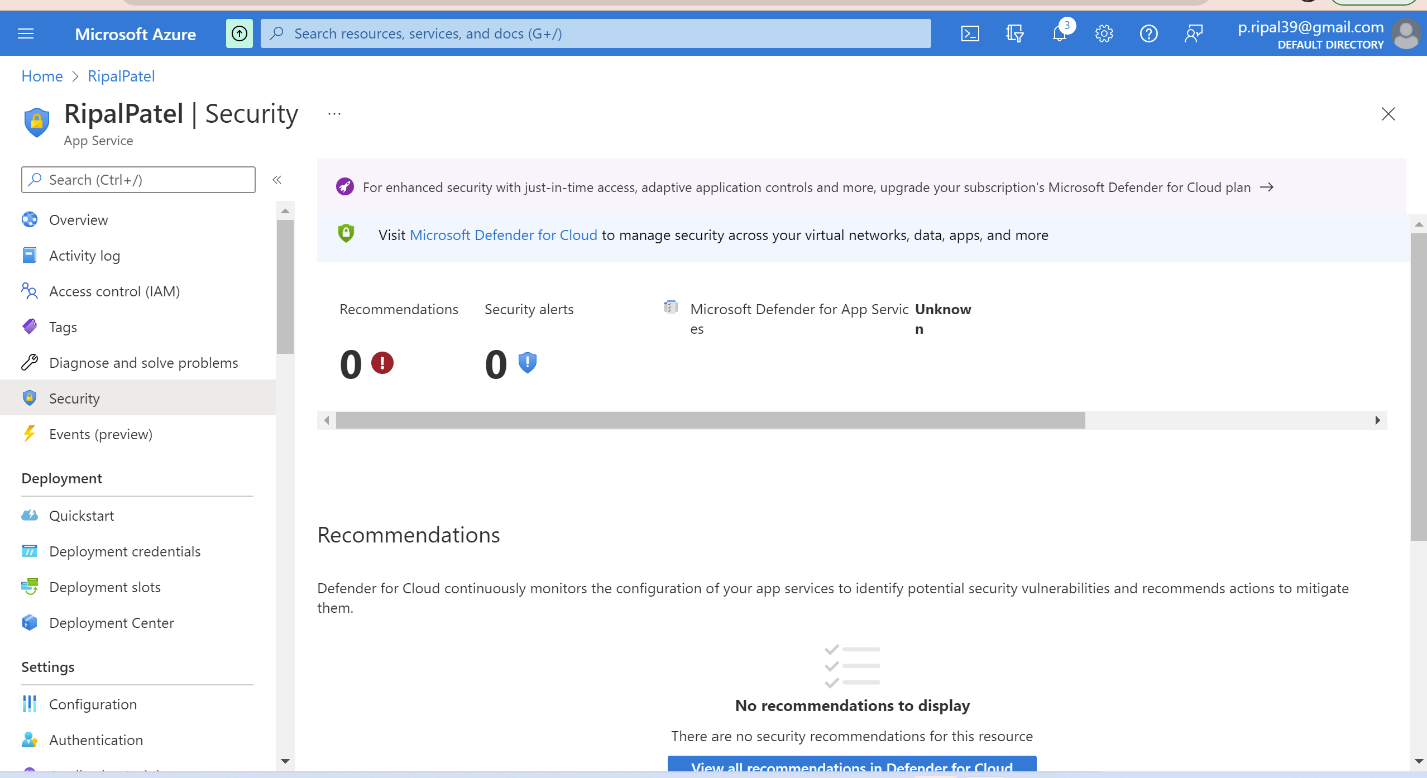
Graphical user interface, application, email

Description automatically generated



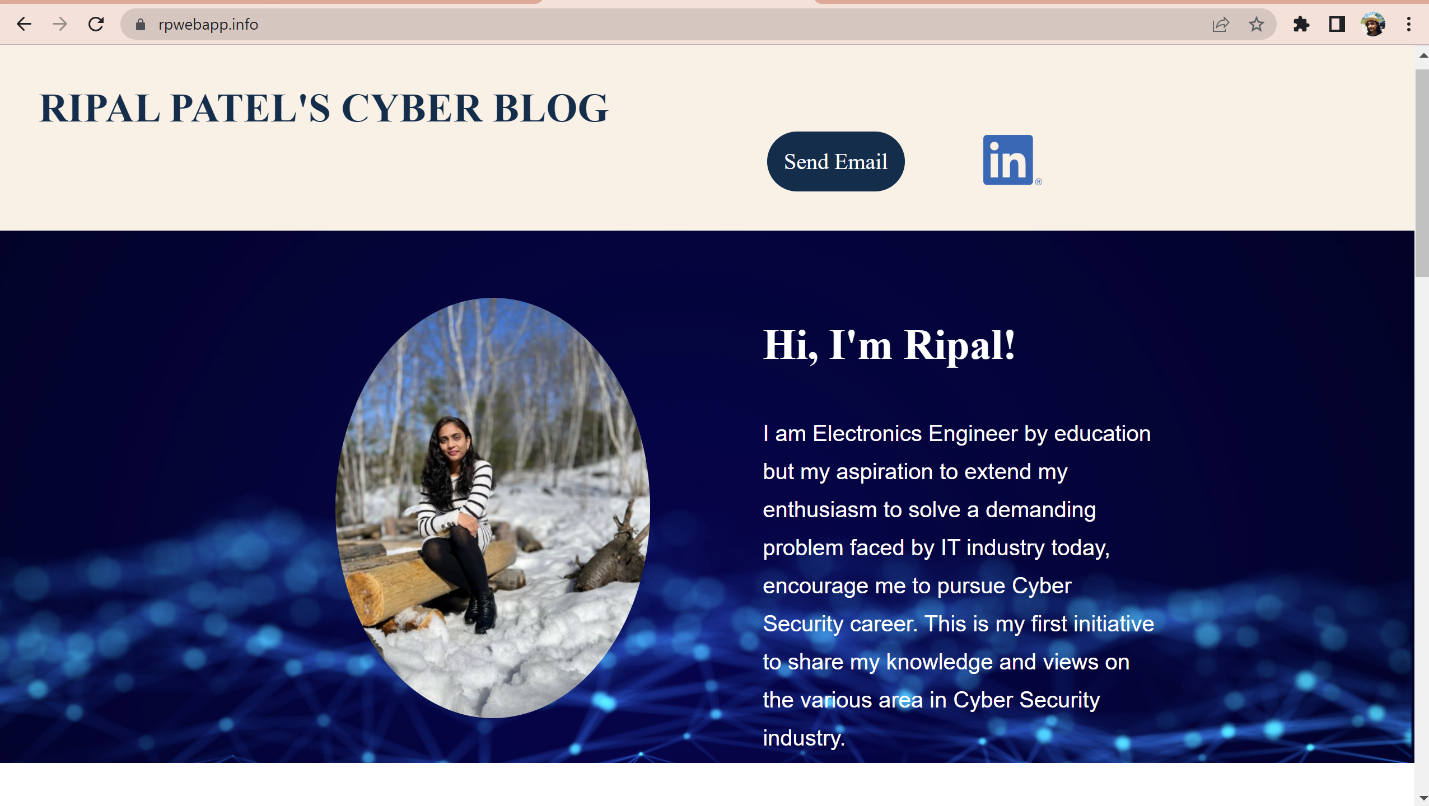
* 1. Your custom rule should now display on the page.

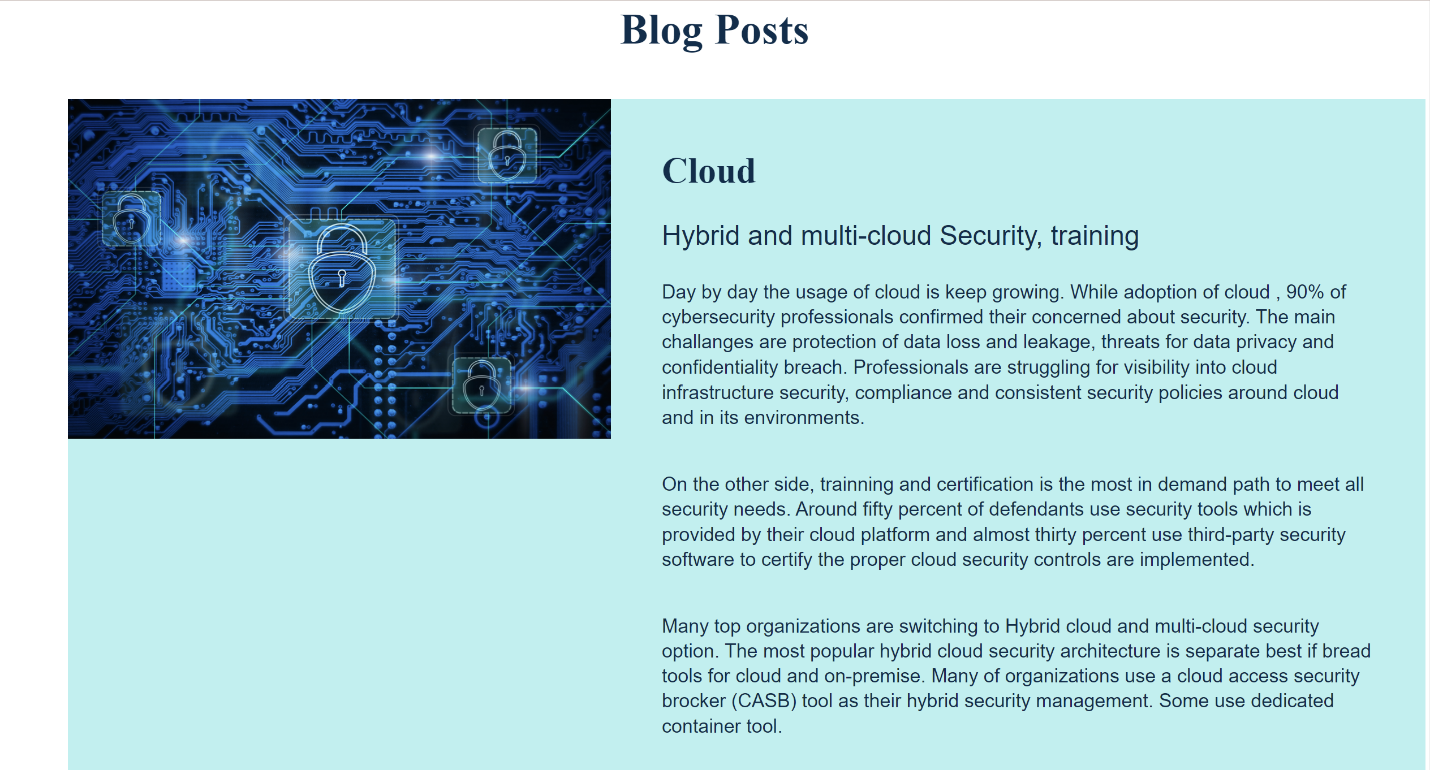


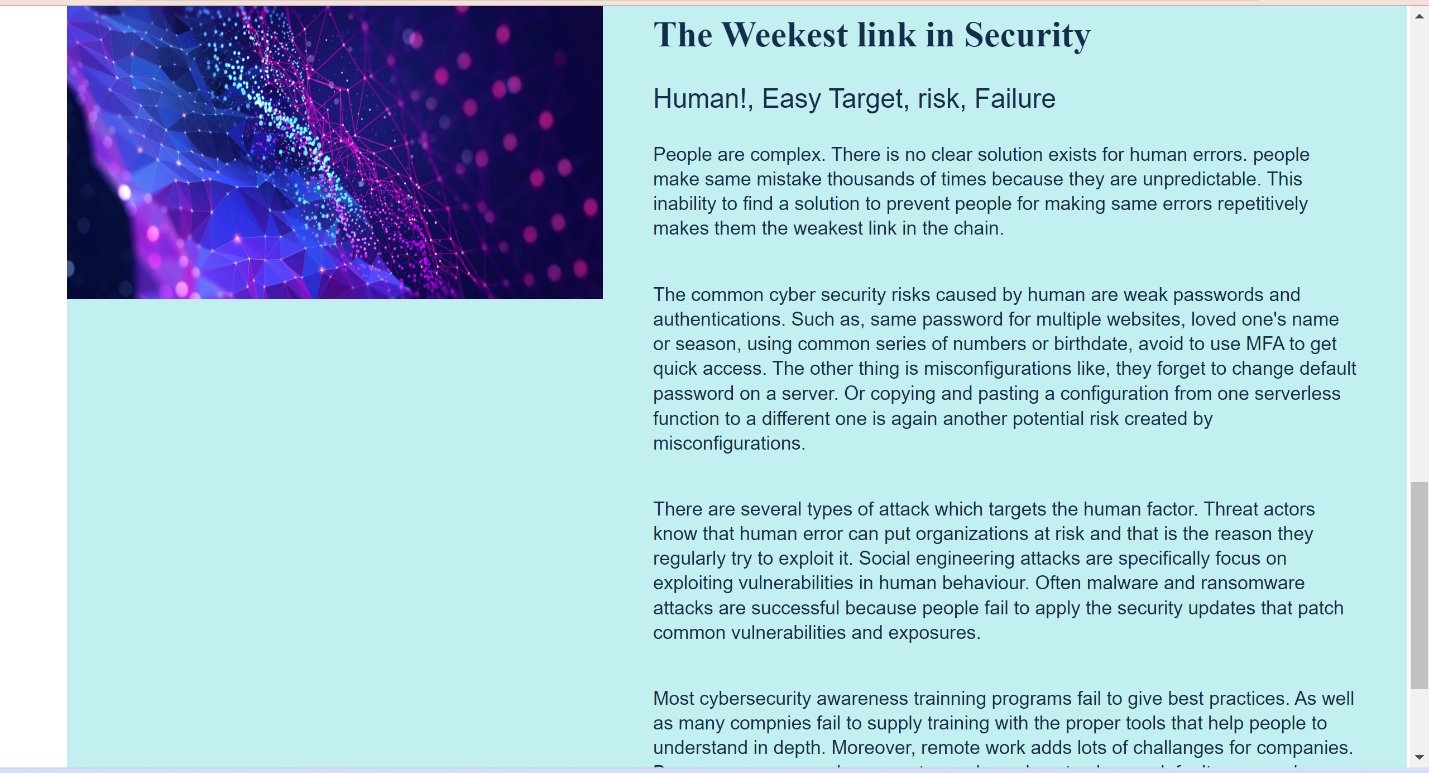


* 1. Congratulations! You have configured the WAF to restrict traffic from accessing your webpage unless the source IP is from the US, Canada, or Australia.

Now Website looks like this.







Certification

