National College of Ireland

Computing

Cyber Security

2020/2021

Mindaugas Prismantas

X17489412

X17489412@student.ncirl.ie

Easy Blocker

Technical Report

Contents

Executive Summary

This project consists of developing secure internet browsing for less tech savvy people. To achieve that it requires using raspberry pi as a secondary router to be able to block domains and DNS from the internet, so that the ads or malicious websites are forever blocked behind the second router filter. All devices that are connected to the ethernet cable are influenced and protected by the raspberry pi shielding more than one device from malicious activities that happen on web. Like a malicious website that throws bunch of pop-up ads that forces you to download a virus. The statistics show that the malware is not going anywhere, and it will not go away any time soon, “560,000 new pieces of malware are detected every day.” *(A Not-So-Common Cold: Malware Statistics in 2021 | DataProt, 2021) [1]*, and the best way to protect IoT devices form malware is to prevent it completely. On this project users are allowed to make their own filters and add their own websites that they wouldn’t want to see, giving them the customisation if needed. There are already 3rd party lists made that get updated frequently and you can choose them without any difficulty. It is an easy integrate for all homes, as you just plug it into the network, and it should be able to work straight away. Making it less of a hassle for people that do not really know how to use computers apart from browsing. This process will help people to have way smoother internet browsing experience without any disturbances. As a cyber security student, I want to make sure that everything is secure, running without any issues. That is why having a secondary router that is isolated on local network is a best way to do it. A sure way that would be a very hard for hackers to tinker with.

1.0 Introduction

1.1. Background

I have taken this project because I wanted to make a safe and browsable space for everyone, either if it is a child or adult or an elderly person. You never know when somebody is going to click a suspicious looking ad or a link that might bring you to a place to download something. I wanted malicious things like this gone as its kind of annoying and it can endanger some one's system on daily basis when browsing the internet. Especially the predatory scams on that elderly people scammed into, like the ones that say your computer is running into issues call this number. I really do not want anything like that on the internet and seeing elderly people being scammed their money is really disheartening. Of course, you could use a simple ad block but not everyone knows about them or understands them enough to know what they do. That is why I thought it is a perfect opportunity to create something that is simple to install into their home without needing any expertise. Just passively running in the background and blocking things that you do not want. I wanted to make something that is unique, do it yourself thing. That nobody else will have it apart from me. There is plenty of competition especially when there's very similar project like Pi-Hole which does the same thing but better as it has a team working behind it. There are also extensions like "uBlock Origins" *(gorhill/uBlock, 2021) [2]* that is widely known, its not the same as my project but it does the job well for an ad blocker.

1.2. Aims

So, the aim of this project is to make a blocker to block domains and DNS on router level. This means that if you have web address or IP address of specific website you can pretty much block it from loading on your browsers. What you do is simply connect raspberry pi to your network and using default settings it should establish the connection between your router and raspberry pi and every device using ethernet cable. Now when browsing the internet on your computer or laptop that is using the ethernet cable you should not be able to see ads. I want it to be as customisable as possible making it very easy to change what you want to block depending on your work. Using an API to upload your own file if wanted or needed. Building a secure API that is only accessible by the owner of raspberry pi. Protected by passwords and accessibility roles. I would not want any unwanted hacker to get into the system and then make an easy backdoor to just come in and snoop around.

1.3. Technology

What technology will you use to achieve what you have set out to do and how will you use it?

What I used for this project was primarily python. Python is very powerful programming language and its really easy to deal with computer-based files that will help me to work with Linux and windows if needed. Raspberry pi to act as a secondary router to make a connection between the devices and the router to block DNS. I will be installing Open-Wrt that is an open-source code that can turn any device into a router, this Open-Wrt is very customisable and provides a lot of options for a router where you are able to install 3rd party packages or even create your own ones.

Django in building the API to communicate between the client and the server and upload a custom file for custom blocking experience. I want it to make it so you do not have to log into the router each time and just access a simple webpage that you are able to upload a file for the router to pick up and once selected it can be used to block customised websites and IP addresses.

1.4. Structure

Provide a brief overview of the structure of the document and what is addressed in each section.

Section 1 – Is an introduction to my project it covers the background and the aims of the project. While technologies are about the technologies that I am going to use in the project and briefly describe what each can do and how I am going to use them.

Section 2 – This section is all about the system of my project. Explaining the projects requirements. Functional requirements will cover all the little bits that I need to make my project come together. It will showcase its security and quality of life features that make my project stand out. Use case diagrams will guide users on how to use my project, how users are able to interact with it and all the system bits that show process how each part communicates with each other. The design and architecture will show off the design model of my project and why I have taken these particular steps as well as architecture behind it. Implementation will be showing off algorithms and functions that are used in the code, to show off features and what each thing does. While graphical user interface is going to represent what the user will see and how will he interact with the system using that GUI. The testing section will explain testing methods and how will be used

to test the project to see if the project is programmed correctly and also the efficiency of the application.

Section 3 – This section covers conclusion; it will give you an overall result of my project.

Section 4 – Further development or research, will be talking about the future and how the project could be elevated and what path would be chosen.

Section 5 – Is all about references, information used to complete this document.

Section 6 – Appendix, this section contains a copy of the original proposal and monthly journals.

2.0 Definitions, Acronyms & Abbreviations

The below explains some of the common terms and language that will be used throughout the document and gives some added context to the details provided.

- **Open-Wrt**
  "The OpenWrt Project is a Linux operating system targeting embedded devices. Instead of trying to create a single, static firmware, OpenWrt provides a fully writable filesystem with package management. This frees you from the application selection and configuration provided by the vendor and allows you to customize the device through the use of packages to suit any application." (Brown, 2021) This firmware is the main point of my project as it allows me to enable my Raspberry Pi into a router which is definitely better than buying an additional router to do the same thing.

- **Pi-Hole**
  "Pi-hole is a DNS sinkhole that protects your devices from unwanted content, without installing any client-side software." (Home, 2021) This project is very similar to mine and you can install this onto raspberry pi or a docker to have network wide ad blocking service.

- **Luci**
  "This is the Open-Wrt "luci"-feed containing LuCI - OpenWrt Configuration Interface." (openwrt/luci, 2021). This interface allows users

to connect to the router and provides a GUI to get all the information needed from that router.

- **Python**
  "Python is a programming language that lets you work quickly and integrate systems more effectively." (Welcome to Python.org, 2021). I used python as it is very powerful programming language. Helped me to build API using "Django"

- **Django**
  "Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel." (The Web framework for perfectionists with deadlines | Django, 2021). Amazing python module that its very easy to get into and use for any python-based project if the developer ever needs an API.

- **OWASP ZAP**
  "OWASP Zed Attack Proxy (ZAP) The world's most widely used web app scanner. Free and open source. Actively maintained by a dedicated international team of volunteers" (The ZAP Homepage, 2021)

- **Boostrap4**
  "Quickly design and customize responsive mobile-first sites with Bootstrap, the world's most popular front-end open-source toolkit, featuring Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful JavaScript plugins." (Mark Otto, 2021). I used to for CSS and style of the Django API.

- **DNS**
  "The Domain Name System (DNS) is the phonebook of the Internet. Humans access information online through domain names, like nytimes.com or espn.com. Web browsers interact through Internet Protocol (IP) addresses. DNS translates domain names to IP addresses so browsers can load Internet resources." (Cloud flare, 2021). Open-Wrt helps me to block out the DNS by looking through all the domain lists.

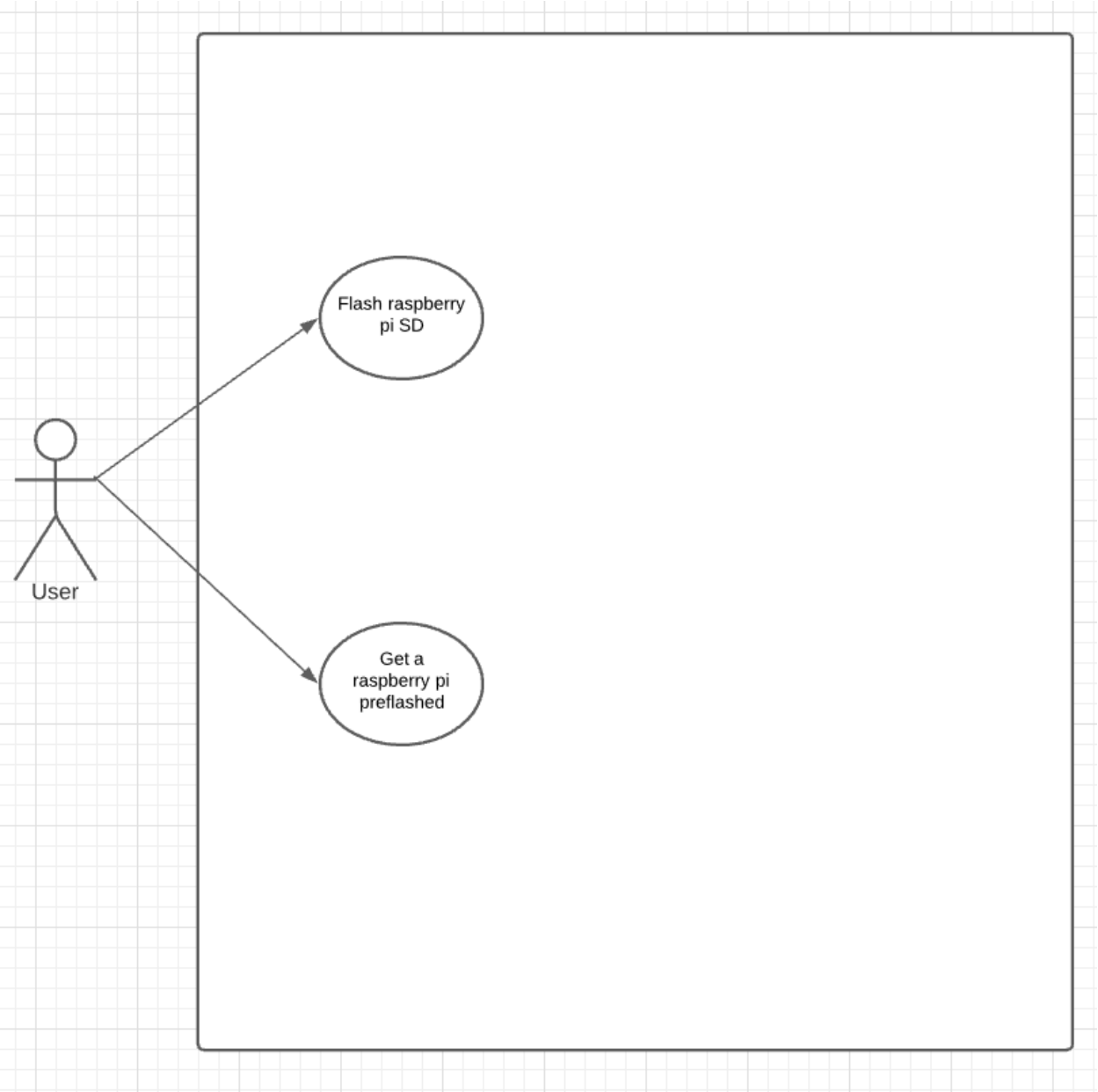
- **API**
  "API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other. Each time you use an app like Facebook, send an instant message, or check the weather on your phone, you're using an API." (What is an API? (Application Programming Interface) | MuleSoft, 2021). I used Django to make an API to communicate between the Open-Wrt and file uploading.

3.0 System

3.1. Requirements

All requirements should be verifiable. For example, experienced controllers shall be able to use all the system functions after a total of two hours training. After this training, the average number of errors made by experienced users shall not exceed two per day.

3.1.1.1. Use Case Diagram



3.1.1.2. Requirement 1 <Preparing the Raspberry Pi>

3.1.1.3. Description & Priority

This use case describes the method of how to flash the SD card.

3.1.1.4. Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

Scope

The scope for this is to have working secondary router as a raspberry pi.

Description

This process is going to explain how to flash the SD card to have fully working open-wrt on a raspberry pi

Use Case Diagram

User

Flow Description

Precondition

Need to have a Sd card, Raspberry Pi imager Open-Wrt image for Raspberry Pi.

Activation

This use case starts once the user has met all the preconditions.

Main flow

1. User connects SD card to computer.
2. User opens Raspberry Pi imager.
3. User then selects operating system and presses custom.
4. User navigates where his Open-Wrt image is and selects it.
5. User then clicks on storage and selects the SD card that he wants to flash.
6. User clicks WRITE.
7. The Raspberry Pi imager installs the Open-Wrt.

7.1 The user is prompted that the installation has been successful.

8. User puts SD card back into Raspberry Pi.

Alternate flow

A1: Invalid SD card

1. User inserts SD card into computer.
2. SD card does not get picked up by computer.
3. User needs to change SD card.
4. The use case continues at position 1 of the main flow.

Exceptional flow

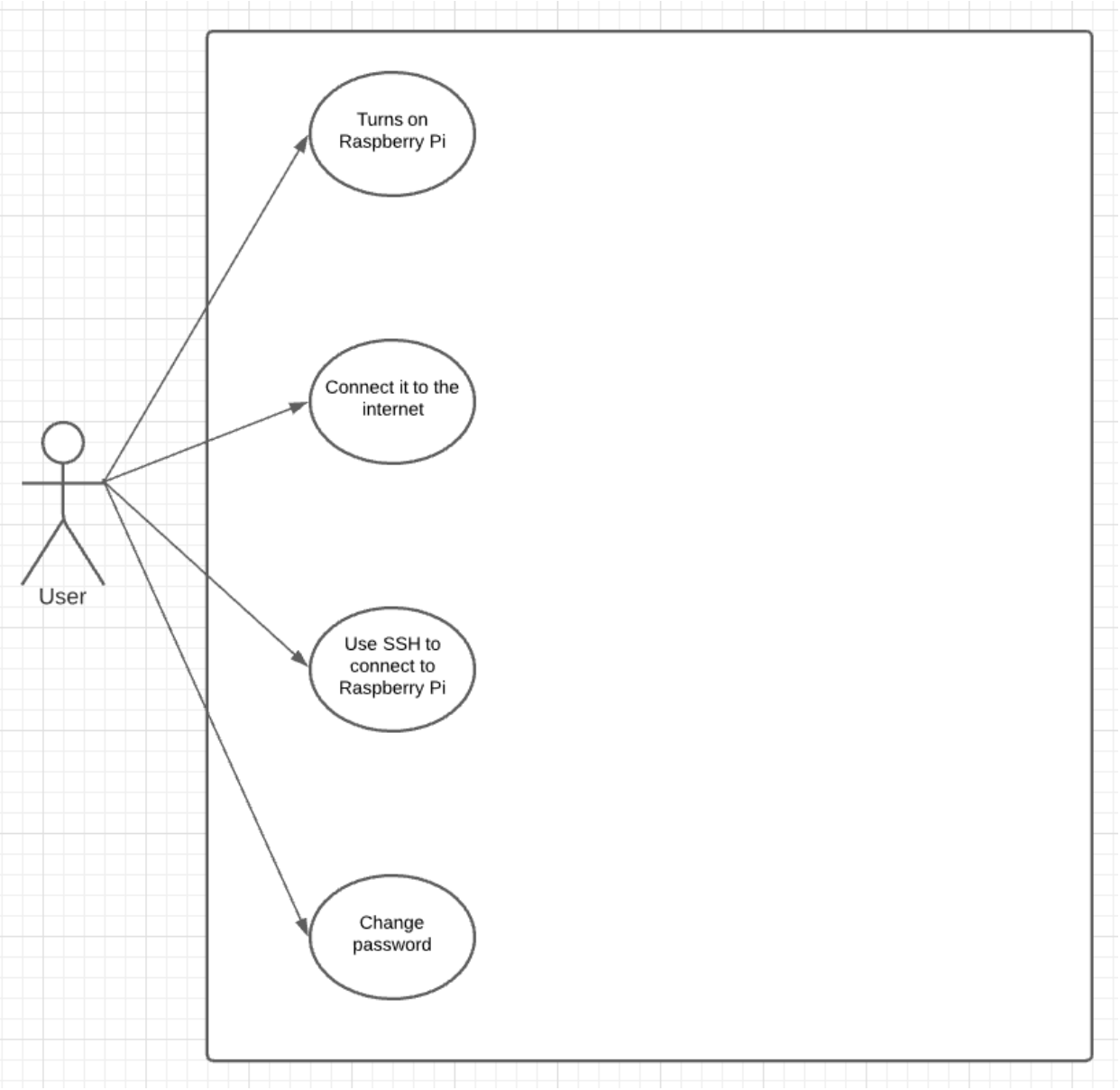
Termination

When the user can see that Open-Wrt logo once the Raspberry Pi is turned on and connected to a monitor.

Post condition

The second router is working and running in the background.

3.1.1.5. Use Case Diagram



3.1.1.6. Requirement 2 <Connecting to the Raspberry Pi>

3.1.1.7. Description & Priority

This use case describes the method of how to connect to the Raspberry Pi.

3.1.1.8. Use Case

Scope

Using SSH connection to get access to Raspberry Pi router.

Description

This process will make the user connect to Raspberry Pi using SSH so the user can start changing things around.

Use Case Diagram

User

Flow Description

Precondition

Raspberry Pi must have an Internet connection via ethernet cable and a computer that's connected to the ethernet cable as well.

Activation

This use case starts once the user has met all the preconditions.

Main flow

1. The user turns on the Raspberry Pi
2. The user connects ethernet cable to the Raspberry Pi
3. The user on the computer uses terminal
4. The user enters the command "SSH"
5. The user enters the "ssh root@192.168.1.1"

5.1 The user should see the open-wrt logo or prompted into root@OpenWrt shell.

6. The user is asked to enter a new password.

6.1 The system checks if the password meets the requirements.

6.2 The system confirms that the password is okay.

Alternate flow

A1: Weak Password

1. The system asks user to enter a new password.
2. User enters a new password.
3. The use case continues from position 6.1.

A2: No internet connection

1. The user tries to ssh to the Raspberry Pi
2. The terminal displays "connection timeout"
3. The use case continues from position 2.

A3: User can use "PuTTY" to access the Raspberry Pi

1. The user turns on "PuTTY" on computer
2. Enters Host name as "192.168.1.1" and Port as "22"
3. User presses open.
4. User enters the login as "root"
5. The use case flow continues from point 5.1.

Exceptional flow

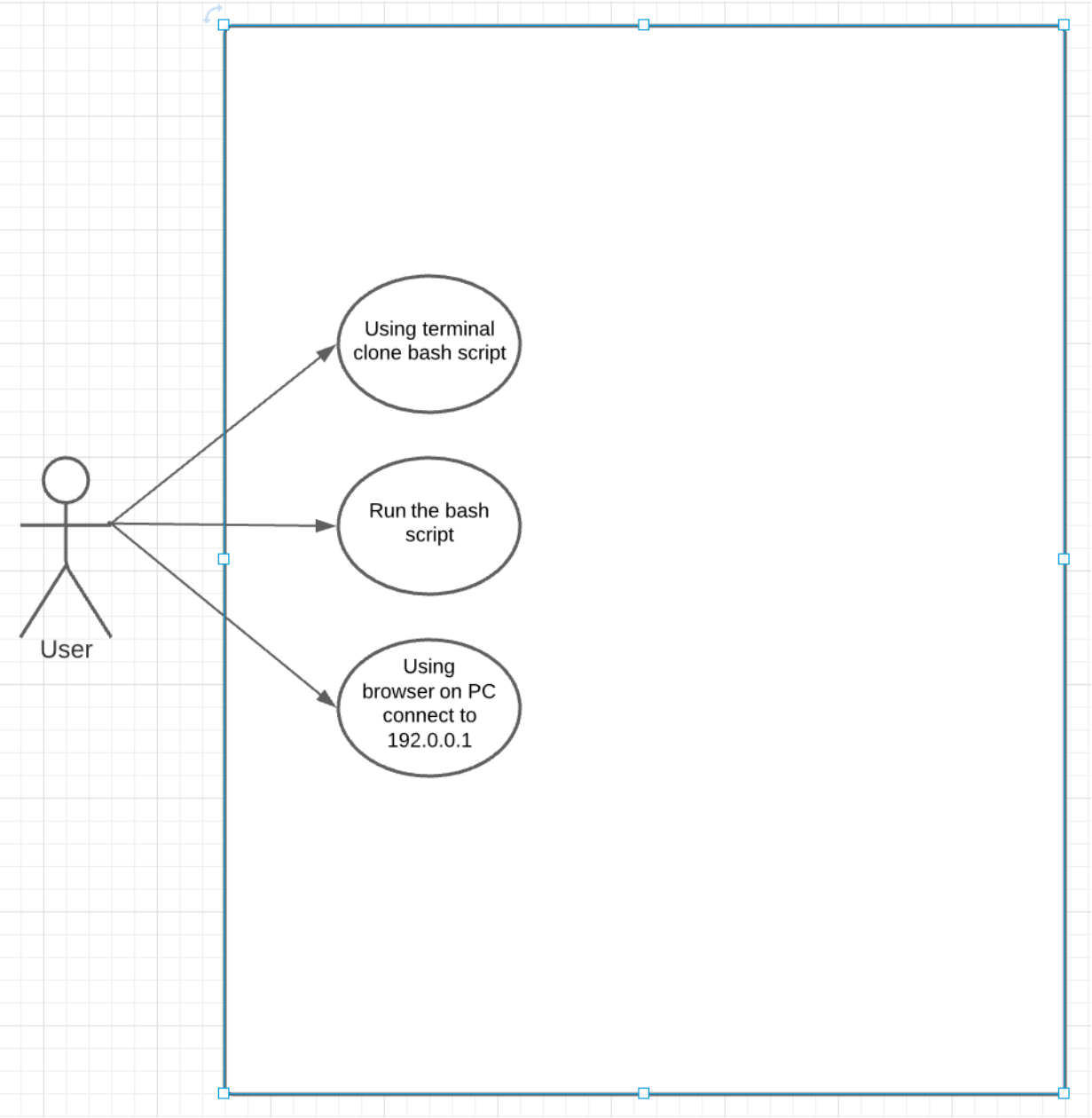
Termination

Once the user is able to see the open-wrt logo the user is ready for next time.

Post condition

Open-Wrt is working and is running in the background but not configured as second router.

3.1.1.9. Use Case Diagram



3.1.1.10. Requirement 2 <Setting up the environment>

3.1.1.11. Description & Priority

This use case describes the method of how to set up the environment

3.1.1.12. Use Case

Scope

The scope of this is to run the bash script to make the connection between the first router and the second.

Description

This process shows the user how to run the script to set up specific requirements for the router to make connection between the routers.

Use Case Diagram

User

Flow Description

Precondition

Raspberry Pi must have an internet connection.

Activation

This use case starts once the user has met all the preconditions.

Main flow

1. User must clone the bash script from the repository.
2. User must enter the command to run the bash script.
3. User must wait for the process to finish.
4. The user needs to restart the Raspberry Pi.
5. The user needs to wait for it to load.
6. The user on the computer should open browser and enter an IP address of “192.168.0.2” to confirm that open-wrt is fully functional.
7. The user must login into the secondary router by using the same credentials as the Raspberry Pi.

Alternate flow

A1: Bash script error

1. The user must check internet connection.
4. User enters a new password.
5. The use case continues from position 6.1.

A2: No internet connection

4. The user tries to connect to “192.168.0.2” to the second router.
5. The browser displays “connection timeout”
6. The use case continues from position 4.

Exceptional flow

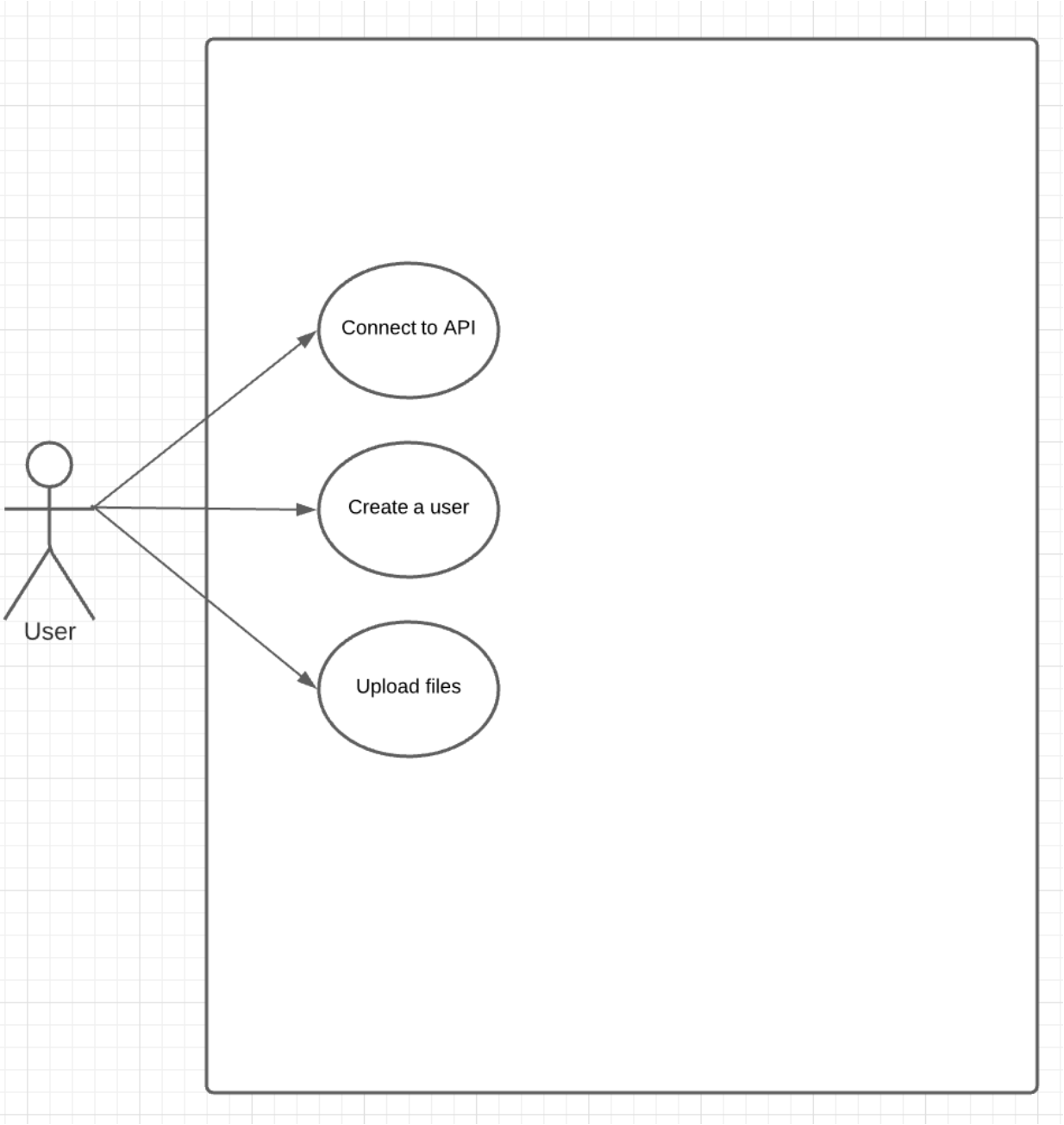
Termination

Once the user can see a login screen and enter their details.

Post condition

Open-Wrt is working and is running in the background but a configured as second router.

3.1.1.13. Use Case Diagram



3.1.1.14. Requirement 3 <Connecting to API>

3.1.1.15. Description & Priority

This use case describes the method of how to connect to API and upload a file.

3.1.1.16. Use Case

Scope

The scope of this use case is to connect to an API and be able to upload and delete files from the data base.

Description

This process should allow user to be able upload a custom file with blocked domains and IPs to block out unwanted content for extra customisation.

Use Case Diagram

User

Flow Description

Precondition

Raspberry Pi must have previous conditions met by running bash script.

Activation

This use case starts once the user has ran bash script.

Main flow

1. The user tries to connect on the browser to 127.0.0.1:8000 to connect to an API.
2. The user can navigate to the menu
3. The user can click upload and upload their own file.

3.1 The user has to enter file name, information about the file and select file which they want to upload.

4. User has to enter file name file has to be in .txt format for it to be read.

Alternate flow

A1: The User can click file page

2. The user clicks file page.
3. The user clicks upload button on a file page.
4. The user is able to enter files name, information about the file and select a file user wants to upload.
5. The use case continues from position 2.

A2: The user wants to delete a file

1. User clicks on files
2. User finds a file he wants to delete
3. User clicks delete button beside the file he wants to delete
4. The use case continues from position 2.

Exceptional flow

E1: No internet connection
1. The user tries to connect to “192.168.0.2” to the second router.
2. The browser displays “connection timeout”
3. The use case continues from position 1.

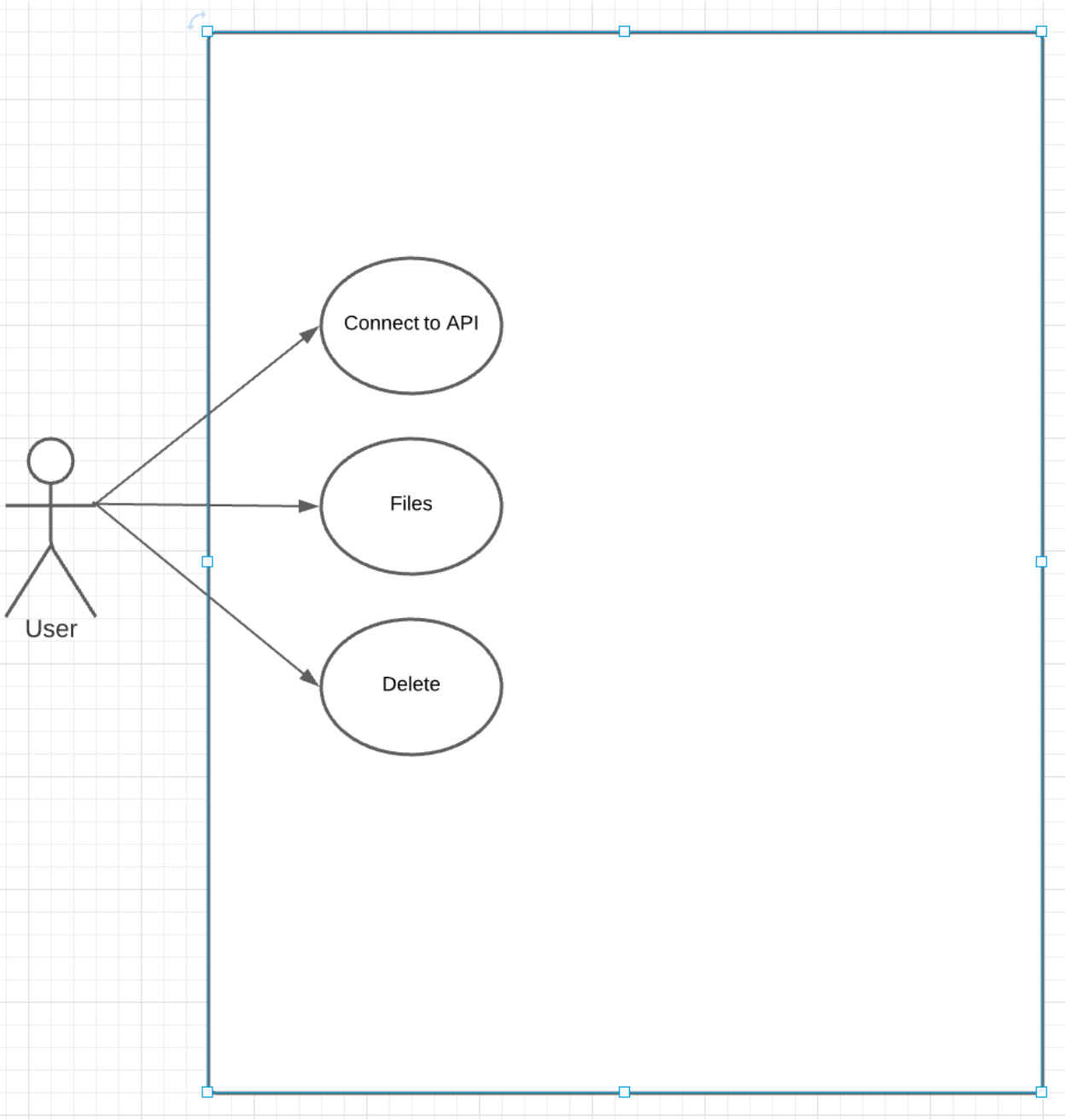
Termination

Once the user uploaded file he is redirected to files page and upload flow stops.

Post condition

The user is now able to log into secondary router and enable their custom filter.

3.1.1.17. Use Case Diagram



3.1.1.18. Requirement 4 <How to delete files >

3.1.1.19. Description & Priority

This use case describes the method of how to navigate and delete files from the database

3.1.1.20. Use Case

Scope

The scope of this use case is to connect to an API, check the database and be able to delete a file from the database.

Description

This process should allow user to be able upload a custom file with blocked domains and IPs to block out unwanted content for extra customisation.

Use Case Diagram

User

Flow Description

Precondition

Raspberry Pi must be up and running

Activation

This use case starts once the user can connect to API using 127.0.0.1:8000

Main flow

1. The user tries to connect on the browser to 127.0.0.1:8000 to connect to an API.
2. The user can navigate to the menu
3. The user can click files.
4. The user can click delete the file from the data base by clicking delete button.

Alternate flow

A1: The User can click Upload page

6. The user clicks upload page.
7. The user is able to enter files name, information about the file and select a file user wants to upload.
8. The use case continues from position 6.1.

A2: The user wants to delete a file

5. User clicks on files
6. User finds a file he wants to delete
7. User clicks delete button beside the file he wants to delete
8. The use case continues from position 2.

Exceptional flow

E1: No internet connection
5. The user tries to connect to “192.168.0.2” to the second router.
6. The browser displays “connection timeout”
7. The use case continues from position 1.

Termination

Once the user deletes a file its prompted back into files page where it displays all files in the database.

Post condition

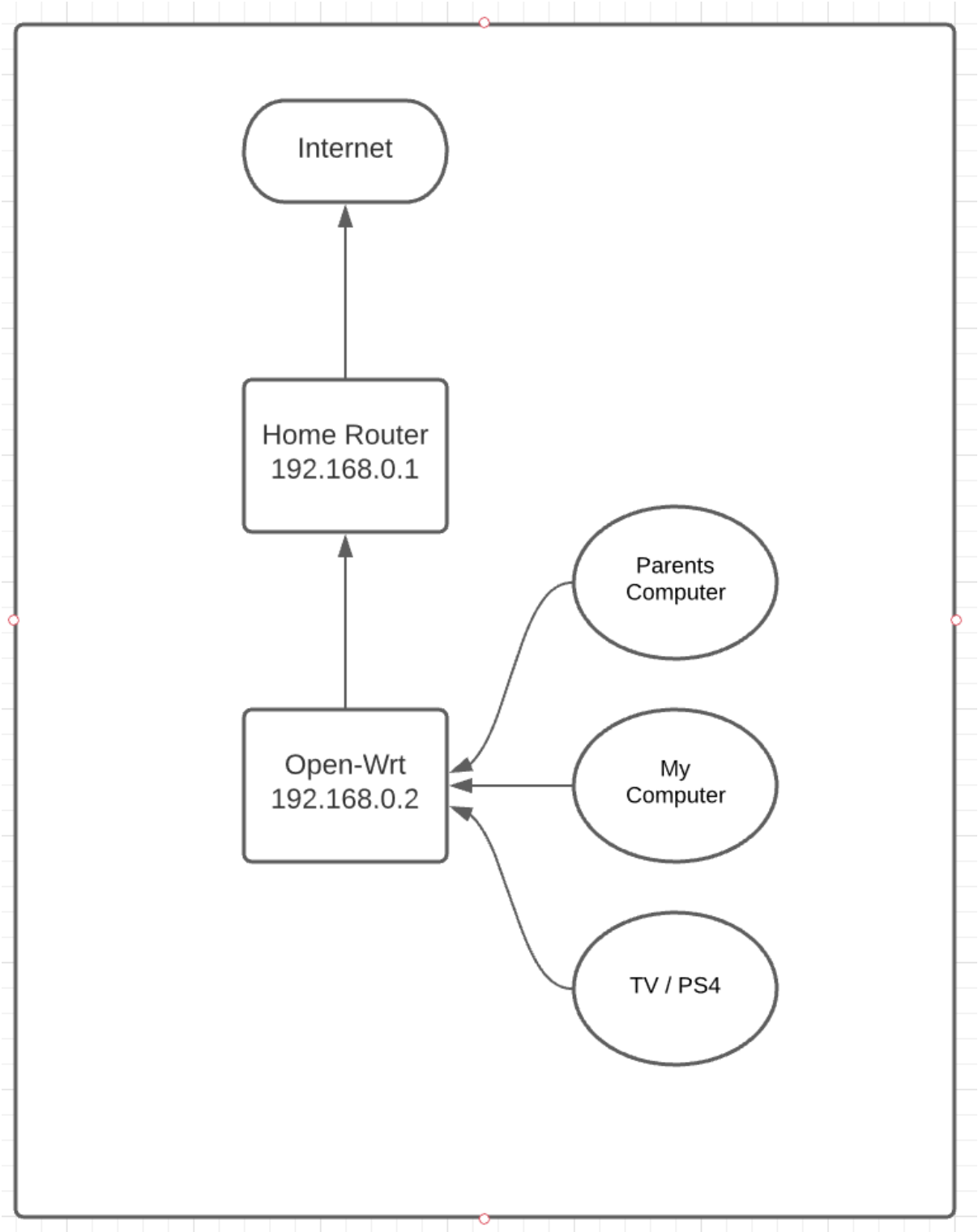
The user can now see that the file they deleted does not appear in the files page of API.

3.2. Design & Architecture

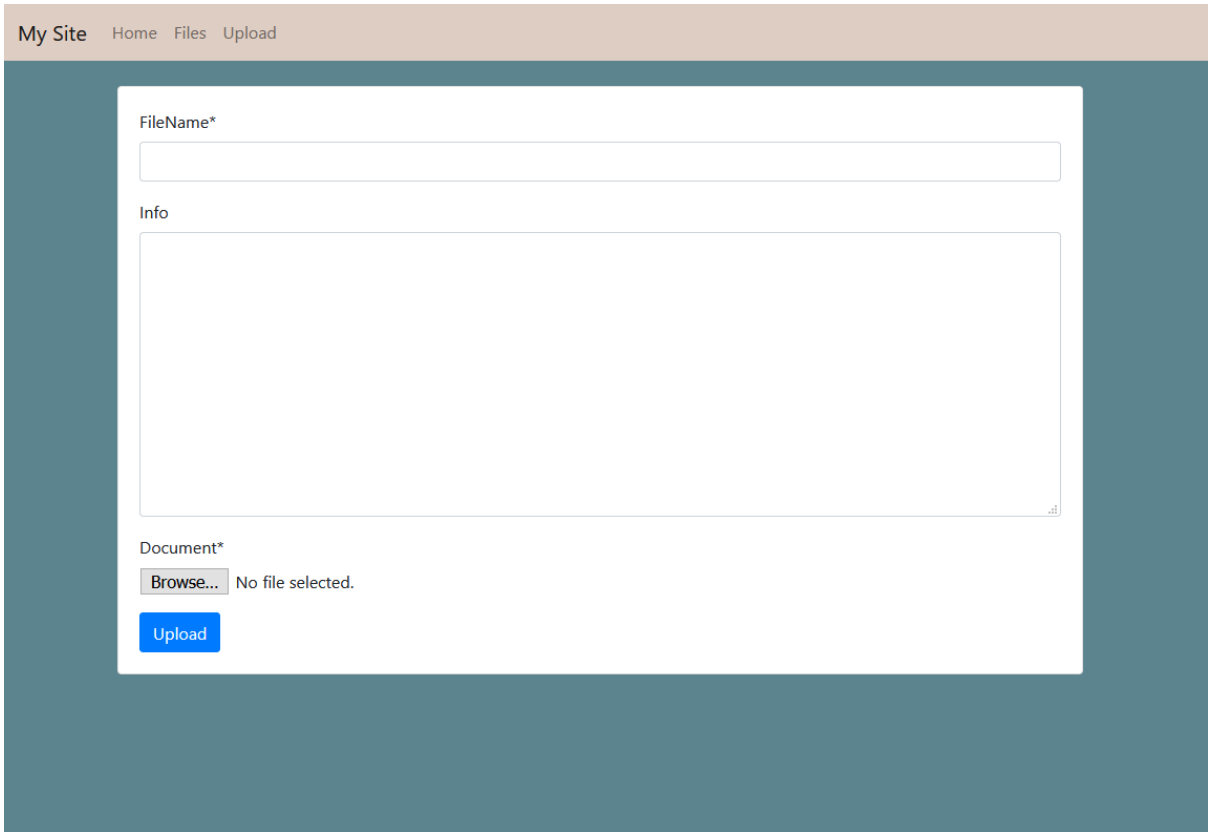
```
config interface 'loopback'
        option ifname 'lo'
        option proto 'static'
        option ipaddr '127.0.0.1'
        option netmask '255.0.0.0'

config globals 'globals'
        option ula_prefix 'fd00:a121:35a3::/48'

config interface 'lan'
        option type 'bridge'
        option ifname 'eth0'
        option proto 'static'
        option netmask '255.255.255.0'
        option ip6assign '60'
        option ipaddr '192.168.0.2'
        option gateway '192.168.0.1'
        list dns '192.168.0.1'
```

1st router (Home router), 2nd router (Open-Wrt)

This sets up your Raspberry Pi router with open-wrt, the bash script sets it up for you automatically but as to understand what is going on its pretty much setting up a network interface that is called lan. This interface uses ‘eth0’ which means that it uses 1st and the only ethernet port that is on Pi. Option proto ‘static’ means that I don’t want the routers address to keep changing after each reset or turn off. Option ipaddr is assigning the Pi’s to 192.168.0.2 so whenever user wants to connect to it and change some settings the address is set and will not change even after reset. The gateway enables the 2nd router route all the traffic into the 1st router, so they can communicate with each other and 2nd router allows the connection to the internet. Otherwise, the Pi router is not able to update or upgrade any of its packages and the adblock would not work.

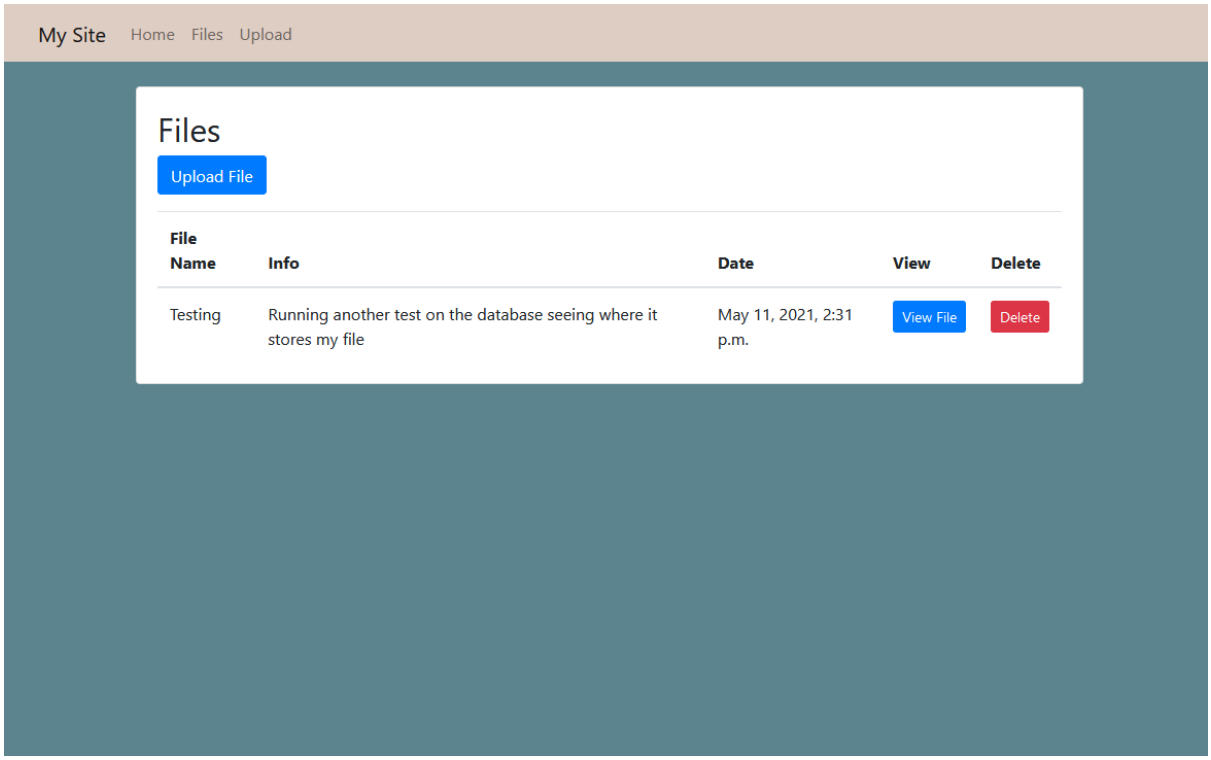


This is the visual representation of how the network is set up after those settings are set up. All the ethernet cable connection on the house gets redirected to Open-Wrt and that allows the ads and malicious links to be blocked once the response comes back through the Open-Wrt router.



For API I chose something simple that everyone would be able to understand and use it very easily. This Upload page to upload a custom filter is easy to use. As you can see from the picture, I chose to have a form layout, Django is amazing for things like this. This allows the user to put in the file name, information about the file and select a file which user wants to upload. The filename and Document for the upload and they are marked with a star (*) and they are required fields to submit the file for the upload.

If the file name is already existing on the database. There is no need to worry for the user as its already covered. What Django does is allows the files with the same name to be uploaded but newer file takes a random letter and number combination and adds at the end of the files name.

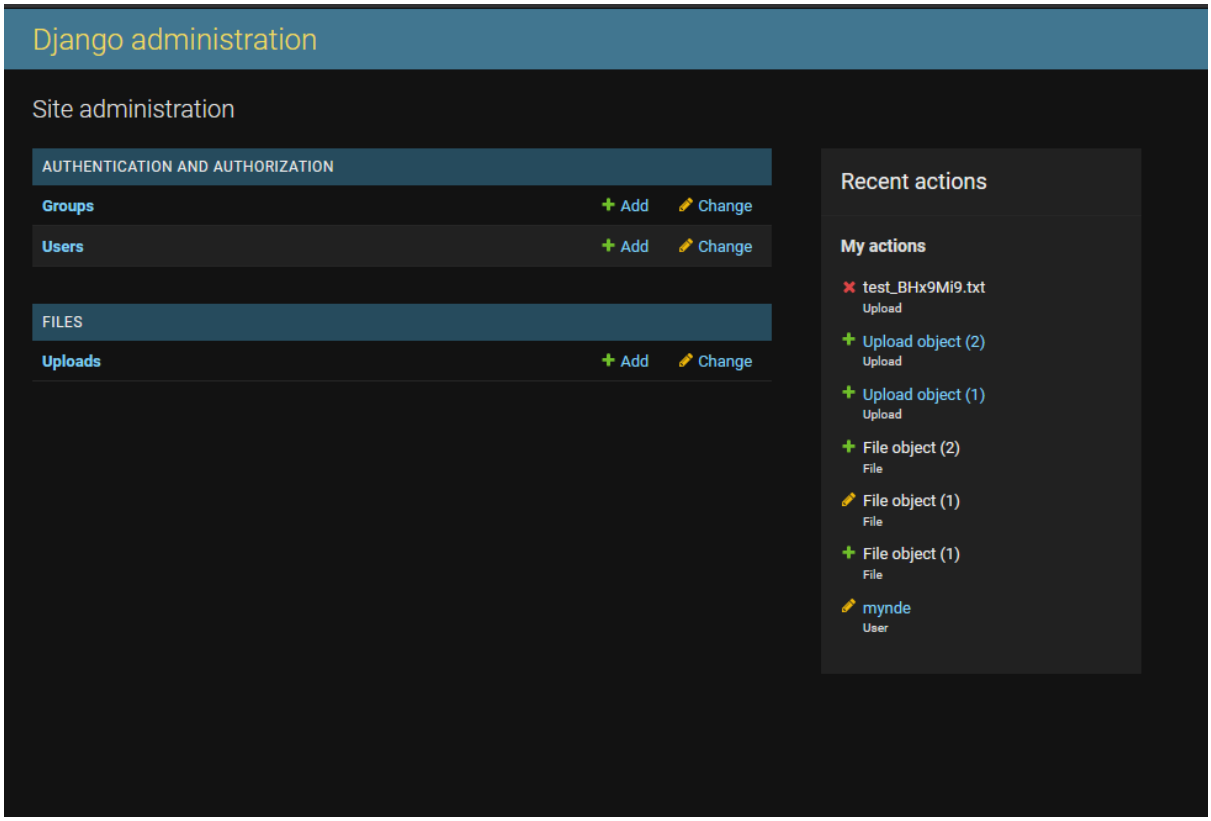


Files page in API displays what files have been upload and stored in the database.

The user can see all the information that they have put into the file, like the file name and information for the file as well as the date that gets uploaded automatically.

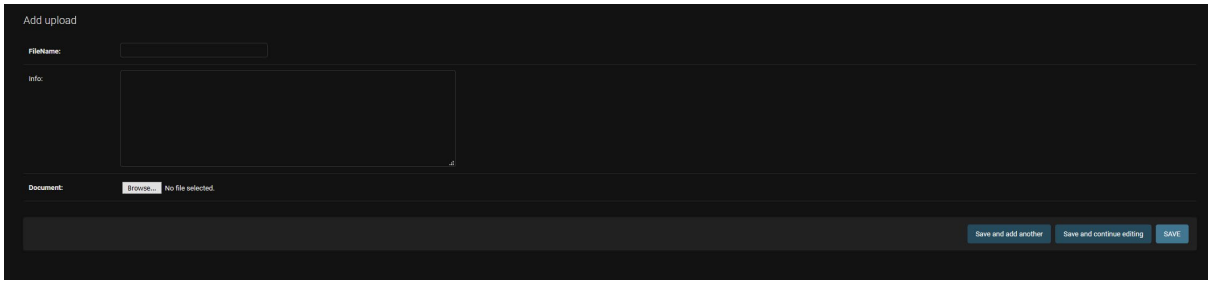
The view button allows the user to check files contents. So, if there is an issue with why custom filter does not work. The user is simply able to just click view file button and see everything that is inside of it.

The delete button allows the user to delete files from the database and its location. So, there can never be left over data unless the user themselves decide not to remove it.

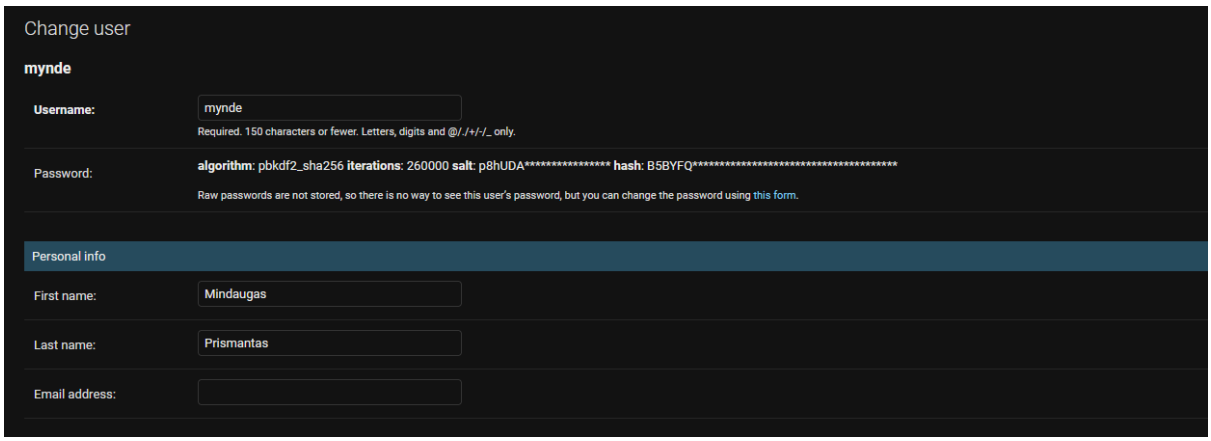


Django come with an administration view; this comes with default Django setup. Here user can see what another user are able to use the API. The user is able to grant access by creating new users and groups. As well as role specific rules on what can and what cannot be done with API of specific user or a group. This provides extra role-based protection that super user is able to create on the go by just accessing “/admin/” page.

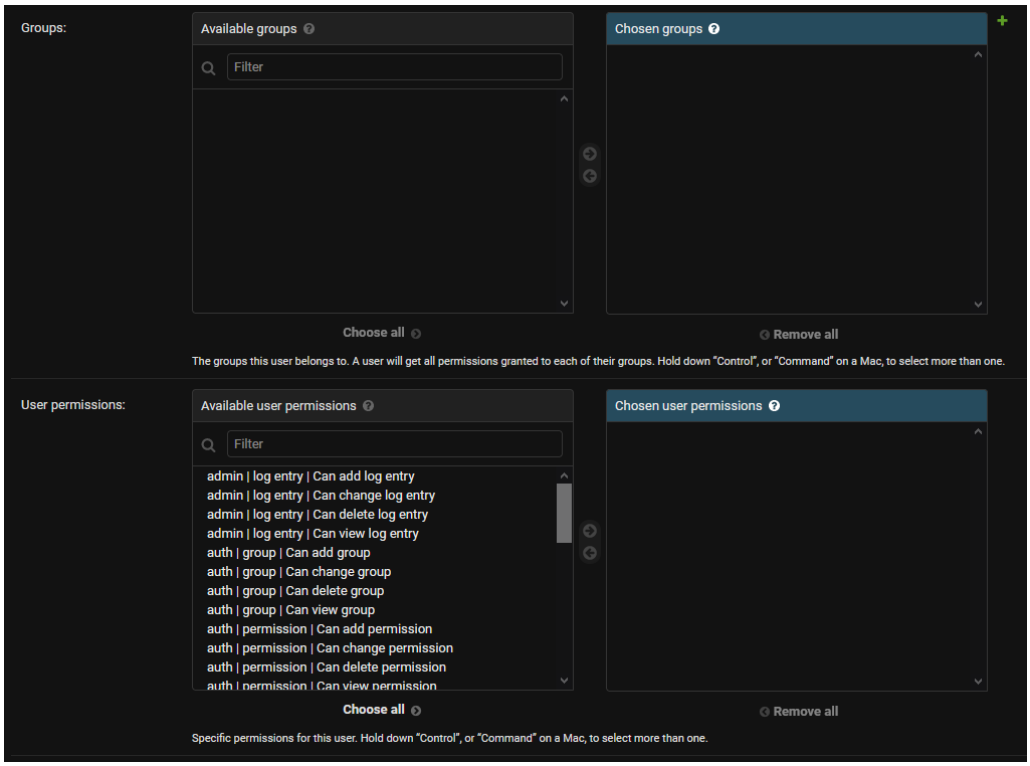
This page also lets the admin to manage the database and if things go south, he can even upload it through there. It is generally not the best practice in doing so as the database can run into issues sooner or later as it is using local upload method instead of API POST methods.



As the picture indicates the process is exactly the same for the upload it reuses the database structure and you're able to upload things from here.



It also password protection, the passwords that are stored for users are in SHA-256 algorithm and with added salt. This keeps the passwords safe and secure. If somebody gains access to the system, the hacker will not be able to just crack your password from accessing this page. Django takes care of the security for developers and if needed there is plenty of more packages that people have created to enhance the security.

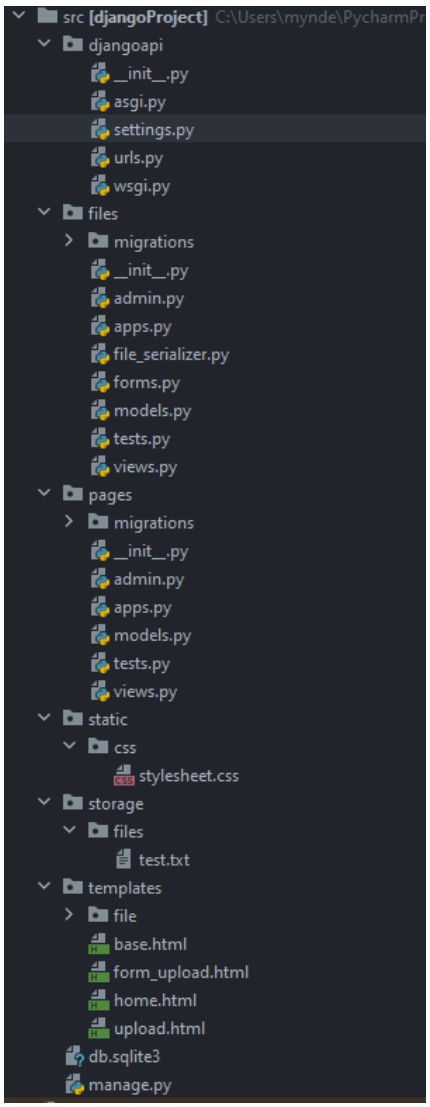


User and group permission are easily customisable, and it uses functions and methods that the developer provides in the code. It gets translated into

this screen and the admin is able to cover certain conditions from happening for normal users and same goes for groups.

For this project there are not any group permission as this API is going to be isolated there is no need to worry about hackers coming and trying to mess with the API.

Django come with really nice features when needing to create an API, the structure of the project is spaced out like this



This structure follows distinctive rules. One of them is for every app there should be a new folder created. For example, in the files and pages folders. Their structure is really similar apart from files having forms and file_serializer. This is due to the fact that files app is responsible with

everything to do with files, like upload and delete, and list them. While the pages app is responsible for display all the methods onto the browser. When making something to do with displaying a page or making new view, all the code will go into pages folder. This is done to make it more easily to distinguish what belongs to what. The less confusion the better for the coder.

If there was another app created something like for customers, anything to do with customers would go inside that folder and when needed to use something from customers it would be referred to that folder.

There's also db.sqlite3 that file is responsible for the database and what is inside of it. It takes from all the apps folders the migrations folders. That is where all the database structure is placed and the rules for said database.

When something is changed on the database, the developer always needs to update the database structure and the rules for it. Django does it after running a command. This only must be done if anything is changed withing the models.py file in the apps folders. The commands are simple: "python manage.py makemigrations" this command verifies the integrity and checks for any changes in the model.py file. If the changes are made Django will make sure to notify the developer if there is actual change and if it is needed to run the second command. The second command is "python manage.py migrate" this is like a commit the rules to the database. It's the file step of verifying the database and the structure.

While manage.py is the brain of Django. That is how the server is start by running this py script and additional parameters like "python manage.py runsever" is an example of how the server starts on the local host.

3.3. Implementation

Describe the main algorithms/classes/functions used in the code. Consider to show and explain interesting code snippets where appropriate.

```
# Create your models here.
class Upload(models.Model):
    fileName = models.CharField(max_length=255)
    info = models.TextField(max_length=100, blank=True)
    document = models.FileField(upload_to='files')
    timestamp = models.DateTimeField(auto_now_add=True)
```

This is the model that creates the database, as displayed in the picture the filename corresponds to the filename in the upload field, info is just information as a text field, and it made so it can be blank. Timestamp does not show up in upload form, but it gets added automatically from the upload time.

Document is made as a file field and it upload the file in the media folder of the database.

```
# Media file location (this is where files go when uploaded)
MEDIA_URL = '/storage/'
MEDIA_ROOT = os.path.join(BASE_DIR, "storage")
```

As the comment says it is a medial file location it located in the project folder. So, it's in "src/storage/" is the default database place and when added with documents file field to upload it into 'files' it becomes "src/storage/files/(filename)" this kind of structure.

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'crispy_forms',

    # My apps
    'djangoapi',
    'files',
    'pages',
]
```

This displays the applications and libraries I have installed. It comes mostly preinstalled with most of it apart from crispy_forms and my apps like "djangoapi", "files" and "pages".

This section controls what can be run in the code so any malicious code could not be executed without being specified here.

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.2/howto/static-files/

STATIC_URL = '/static/'
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')
STATICFILES_DIRS = [
    # os.path.join(BASE_DIR, '/static/')
    BASE_DIR / "static"
]
```

This is for static folder, I used this for placing stylesheet file just to load it as static as I do not need any change on it.

```
def upload_view(request):
    if request.method == "POST":
        form = UploadForm(request.POST, request.FILES)
        if form.is_valid():
            form.save()
            return redirect('file_list')
    else:
        form = UploadForm()
    return render(request, 'form_upload.html', {
        'form': form
    })
```

This code is responsible for showing the upload_view page. It takes in only POST requests, as uploaded file or anything to do with data manipulation should be used in the POST request and not GET request.

It uses form and if the form is valid all the conditions have been completed. It is marked as valid and then saved. Then the user is redirected into file_list or in other words file page.

Otherwise, nothing gets uploaded and the form resets if something fails.

```
def file_list_view(request):
    file = Upload.objects.all()
    context = {
        'object': file
    }
    return render(request, "file/detail.html", context)
```

This code is responsible for displaying files page. This displays all the objects from the database using objects.all() function.

```python
class UploadForm(forms.ModelForm):
    class Meta:
        model = Upload
        fields = ('fileName', 'info', 'document')
```

This is responsible for serialising the model of the database. This is used to translate model code into form so that upload_form could be used to upload file.

```python
def delete_file(request, pk):
    if request.method == 'POST':
        file = Upload.objects.get(pk=pk)
        file.delete()
    return redirect('file_list')
```

This code allows the user to delete the file from the database. It is done with only using POST request. It looks through the files primary key. This is to ensure that the data is getting deleted and not just deleted from the view. Once the file is deleted, it returns the user back to the list.

```python
urlpatterns = [
    path('', home_view),
    path('home', home_view, name='home'),
    path('upload/', upload_view, name='upload'),
    path('file/', file_list_view, name='file_list'),
    path('file/<int:pk>/', delete_file, name='delete_file'),
    path('admin/', admin.site.urls)
]
```

This code is responsible for all the paths to access the webpages. It takes in the path that I would want the API to redirect me, like “upload/” or “file/” then it takes in the view code like upload view from previous pictures and a name that you can refer to in previous pictures also. It allows for a quick reference throughout the code without any hassle.

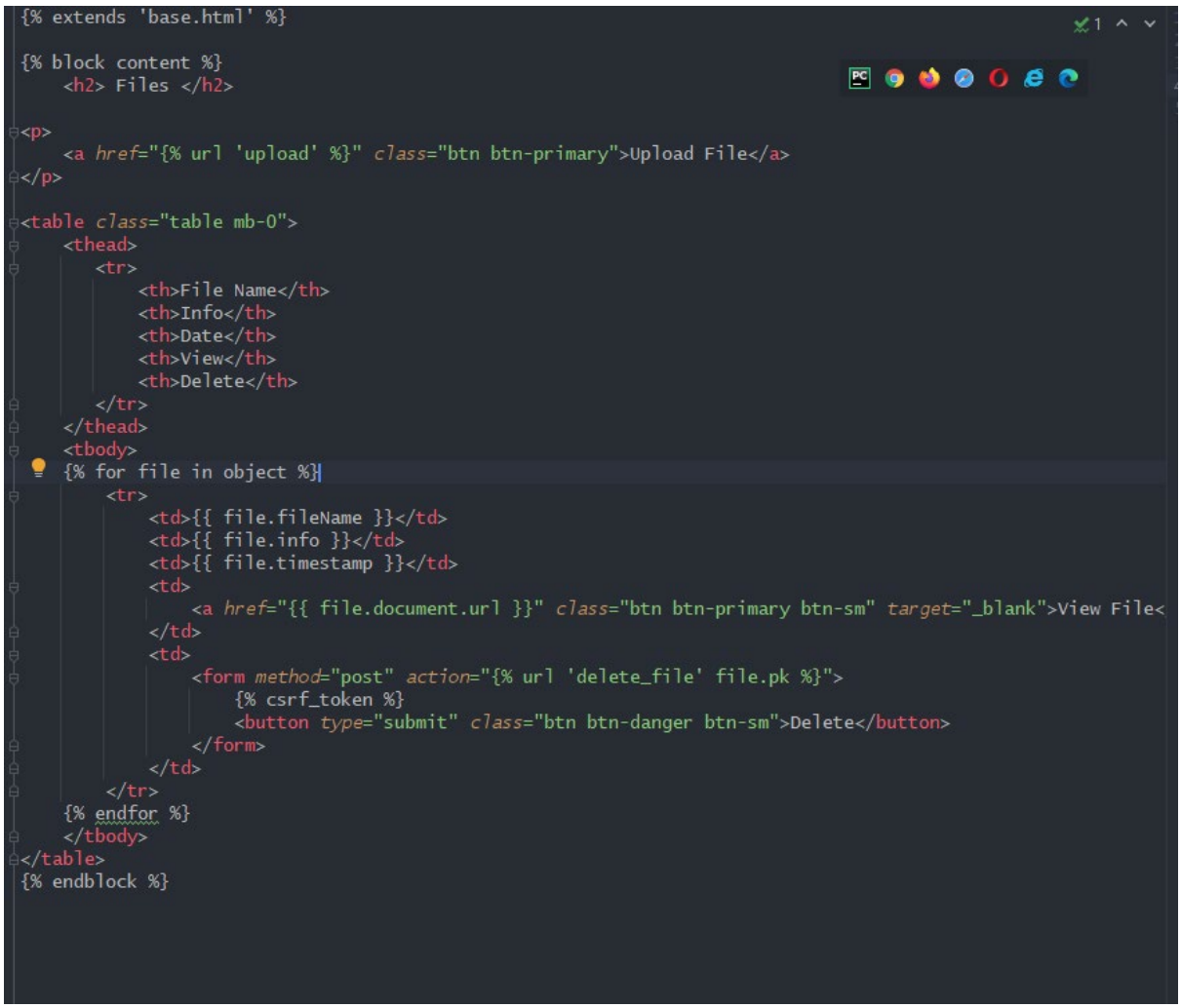
```
{% load static %}
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXF
  <link rel="stylesheet" type="text/css" href="{% static '/css/stylesheet.css' %}">
  <title>{% block title %}My Site{% endblock %}</title>
</head>
<body>
  <nav class="navbar navbar-expand-lg navbar-light mb-4" style="background-color:#decdc3">
    <div class="container">
      <a class="navbar-brand" href="{% url 'home' %}">My Site</a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav">
          <li class="nav-item">
            <a class="nav-link" href="{% url 'home' %}">Home</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="{% url 'file_list' %}">Files</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="{% url 'upload' %}">Upload</a>
          </li>
        </ul>
      </div>
    </div>
  </nav>
  <div class="container">
    <div class="row justify-content-center">
      <div class="col-lg-10 col-md-12">
        <div class="card mb-4">
          <div class="card-body">
            {% block content %}
            {% endblock %}
          </div>
        </div>
      </div>
    </div>
  </div>
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossor
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSn
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/
</body>
</html>
```

This code is typical html code with a bit of Django structure. This allows me to quickly reuse the code if its ever needed. What Django uses is “{%%}” fields like load static to load stylesheet files by specifying it and then using URL for it. This is called base.html and it surrounds “{% block content %}” and “{% endblock %} this is where actual pages code goes into.

```
{% extends 'base.html' %}

{% load static %}

{% block content %}
  <form method="post" enctype="multipart/form-data">
    {% csrf_token %}
    <input type="file" name="myfile">
    <button type="submit">Upload</button>
  </form>

  {% if uploaded_file_url %}
    <p>File uploaded at: <a href="{{ uploaded_file_url }}">{{ uploaded_file_url }}</a></p>
  {% endif %}

  <p><a href="{% url 'home' %}">Return to home</a></p>
{% endblock %}
```

As you can see from the picture this view is being extended from base.html the image above. It allows me to have consistent design of the page and I can reuse this code extension to every single view as long as I don’t plan to make it any different from the other views. It loads stylesheet too and then we have code inside block content. For this case it is for uploading the file,

so it only takes in POST method and takes in CSRF token for verification. As well you can have if statements that is similar to JavaScript.



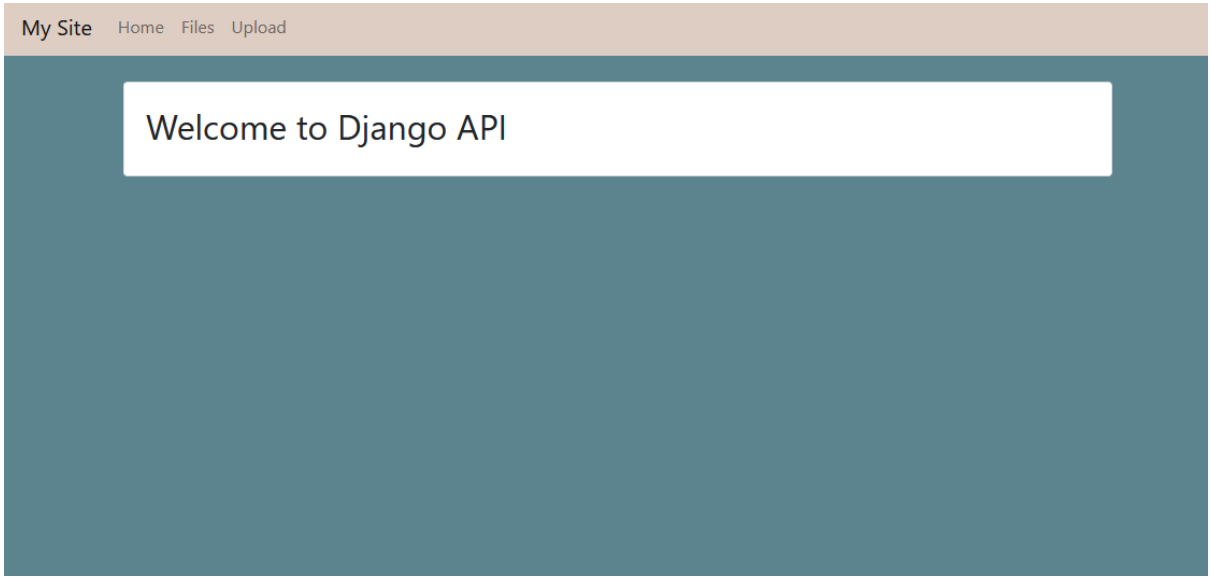
```
{% extends 'base.html' %}

{% block content %}
    <h2> Files </h2>

<p>
    <a href="{% url 'upload' %}" class="btn btn-primary">Upload File</a>
</p>

<table class="table mb-0">
    <thead>
        <tr>
            <th>File Name</th>
            <th>Info</th>
            <th>Date</th>
            <th>View</th>
            <th>Delete</th>
        </tr>
    </thead>
    <tbody>
    {% for file in object %}
        <tr>
            <td>{{ file.fileName }}</td>
            <td>{{ file.info }}</td>
            <td>{{ file.timestamp }}</td>
            <td>
                <a href="{{ file.document.url }}" class="btn btn-primary btn-sm" target="_blank">View File<
            </td>
            <td>
                <form method="post" action="{% url 'delete_file' file.pk %}">
                    {% csrf_token %}
                    <button type="submit" class="btn btn-danger btn-sm">Delete</button>
                </form>
            </td>
        </tr>
    {% endfor %}
    </tbody>
</table>
{% endblock %}
```

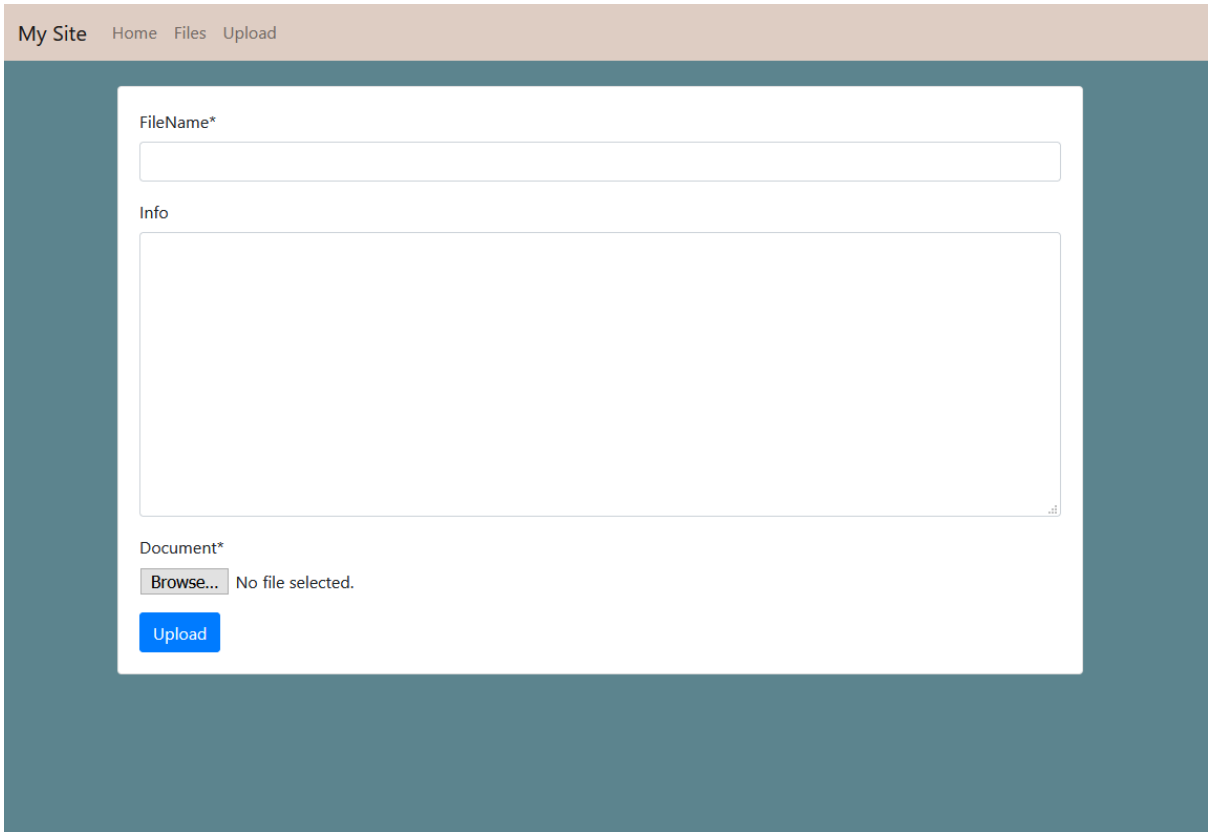
Again, for file page I am using similar structure just different content in the block content. I made a function to display all the objects from the data base and then as you can see, I made a loop function to be able to display each part of the file upload, like name, info and timestamp.

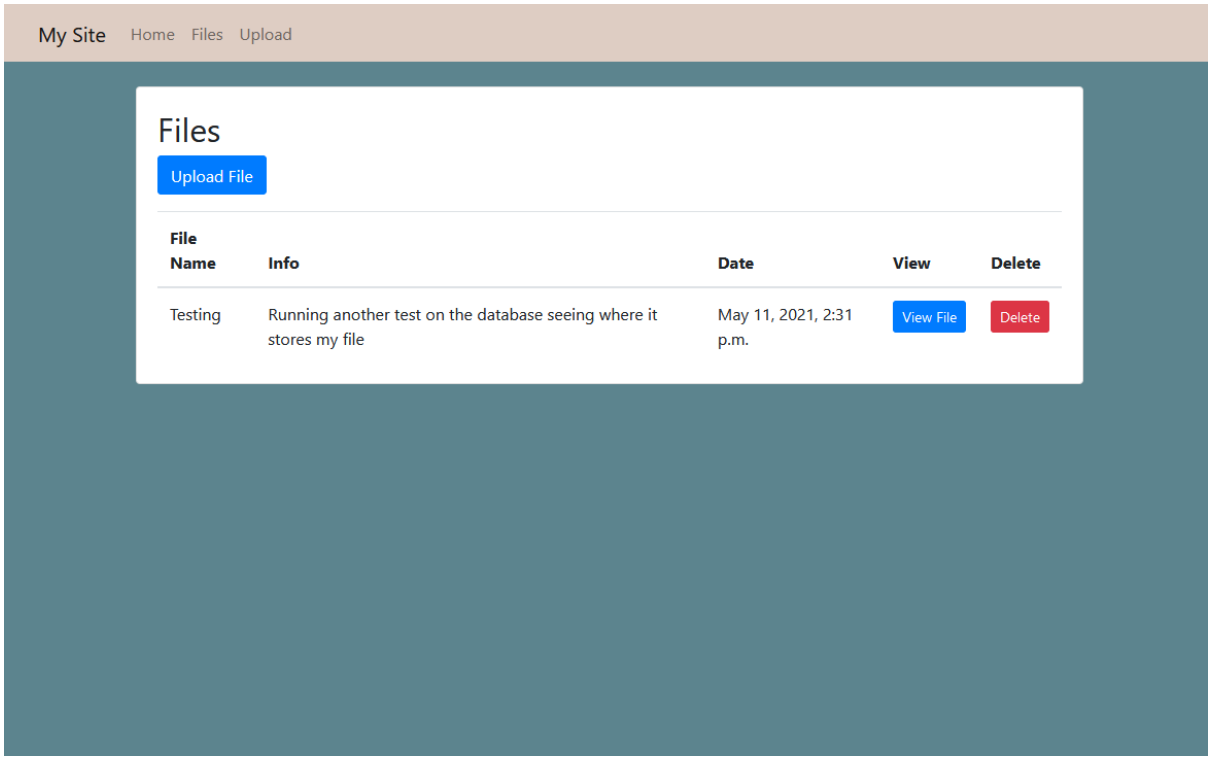
3.4. Graphical User Interface (GUI)

Provide screenshots of key screens and explain what can be seen in each one.

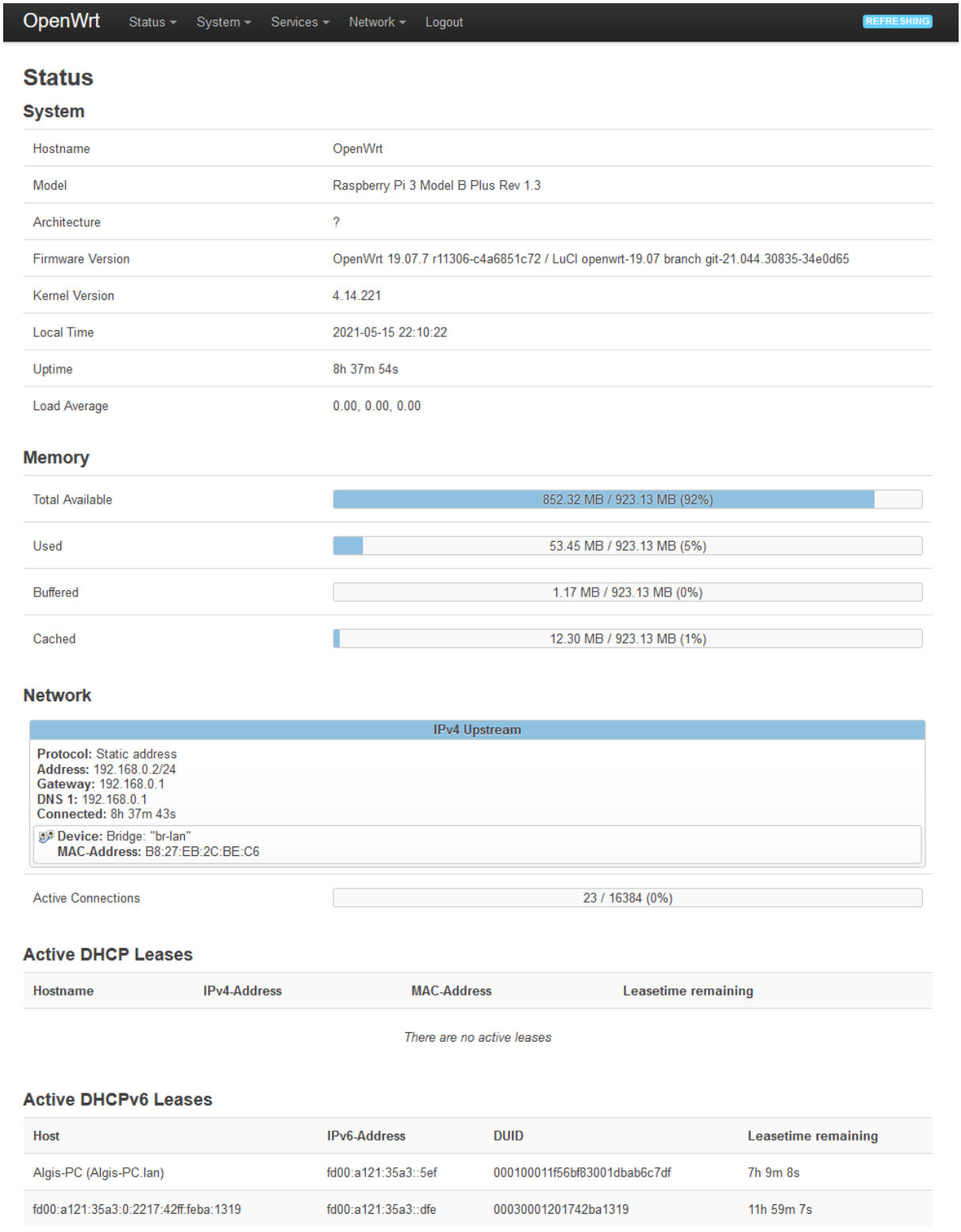
This is the first screen that user sees once he connects to the API using the localhost:8000. On the home page it provides a simple welcome nothing too personalised as this APIs job is just to upload the files.



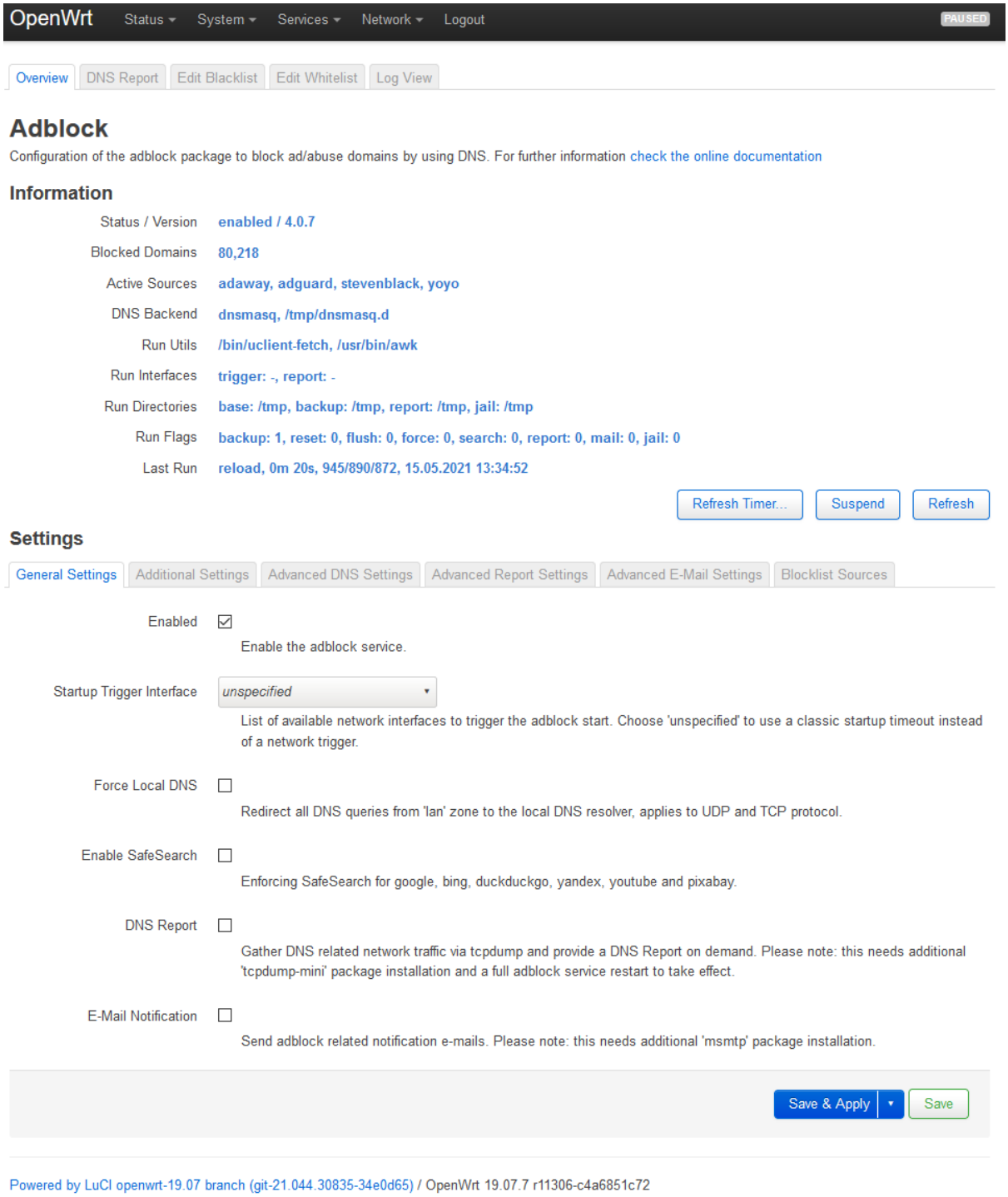




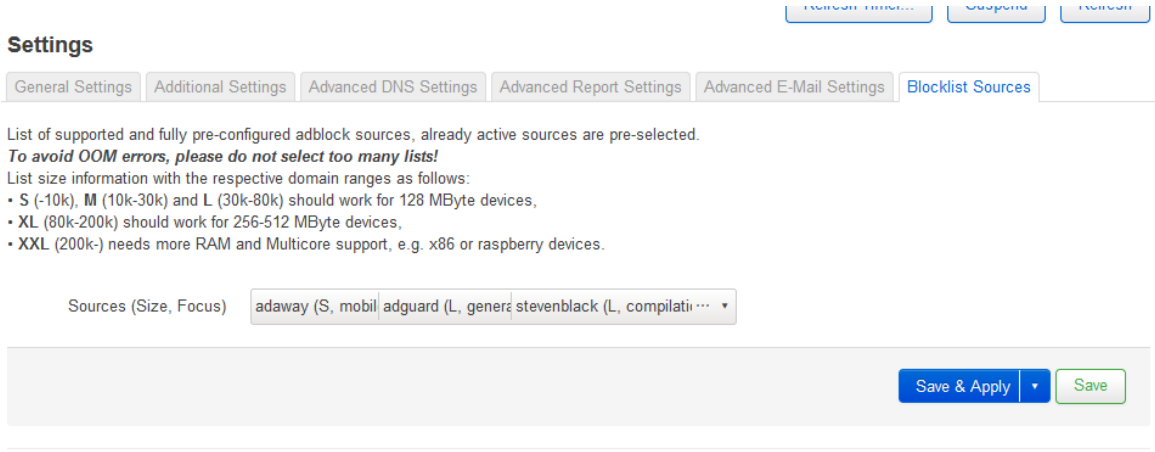
This screen as seen from previous pictures are my API screen, they are pretty basic just to make them as simple as possible without confusing the user.



This is the front page of the Open-Wrt I have not made this screen it comes defaulted with the Open-Wrt image if you selected with “luci-ui”. It displays basic information that could be useful when trying to implement additional things as it shows memory information, the firmware version, active connections and so on.



This page is responsible for AdBlock. As you can see the page displays valuable information for the AdBlock. It shows how many blocked domains it is using, the sources of those domain lists, all the run directories that are important if you want to tinker with the AdBlock itself.



This is where the user can select different source list for the domain blocking. There is already plenty and the more you have the more powerful router you need to have. It is recommended if you want to run list like XXL that contains more than +200 thousand domains, user will need more RAM than the usual routers can provide so it is a little beneficial to run Open-Wrt on Raspberry Pi.

3.5. Testing

Describe any testing tools, test plans and test specifications used in the project. Provide evidence for and results of all Unit, Integration and End User testing that is carried out.

Test Plan:

This section will show you different test that has been carried out while working on this project. It will be a performance test on how quickly websites managed to load comparing the technologies. It will also have a little demo view of how the ads are handled for each technology. Some testing of my own where I run my project for a while and give opinion of what have I noticed and what it feels like when using.

Load Performance Test:

The testing method that will be used is going to be using browser's own monitoring tool. I can record the performance when going into the websites and so on. I will do a test with my project on, then one without my project running and one where its running ad block extension. I will create a table and mark the things that been used for the test and how long it took for the website to fully load, compared to each other.

	IrishTime.ie	Rte.ie	yahoo.com	cnn.com	foxnews.com	espn.com	weather.com	tmall.com	accuweather.com	kiplinger.com	finance.yahoo.com	newyorkfamily.com
Easy Blocker	2.32 s	6.15 s	4.51 s	3.28 s	10.15 s	10.14 s	5.11 s	Failed	4.52 s	1.57 s	11.13 s	5.06 s
uBlock Origin + Nano Defender	2.62 s	5.81 s	3.01 s	7.82 s	11.63 s	10.27 s	5.69 s	Failed	4.58 s	2.21 s	12.54 s	5.30 s
Ublock Origin	3.15 s	5.17 s	2.82 s	8 s	10.32 s	15.08 s	5.43 s	5.01 s	5.16 s	4.13 s	13.13 s	8.31 s
Nothing	6.36 s	4.97 s	8.48 s	15.11 s	11.63 s	20.78 s	6.36 s	15.93 s	7.52 s	9.57 s	23.99 s	10.63 s

Load Times
1 - 5 s
5- 10 s
10 - 15 s
15 - 20 s
20 - Failed

As you can see from the picture each website has different loading times, as they have different functions and ways to load the data onto the page. I have taken extra steps and ran these tests more than once and got the media load times of all of them using each of the technologies or none of them at all.

In this comparison you can clearly see that most of the time loading websites with nothing is slower than actually using something like ad block, so you can already understand that these extensions and easy blocker

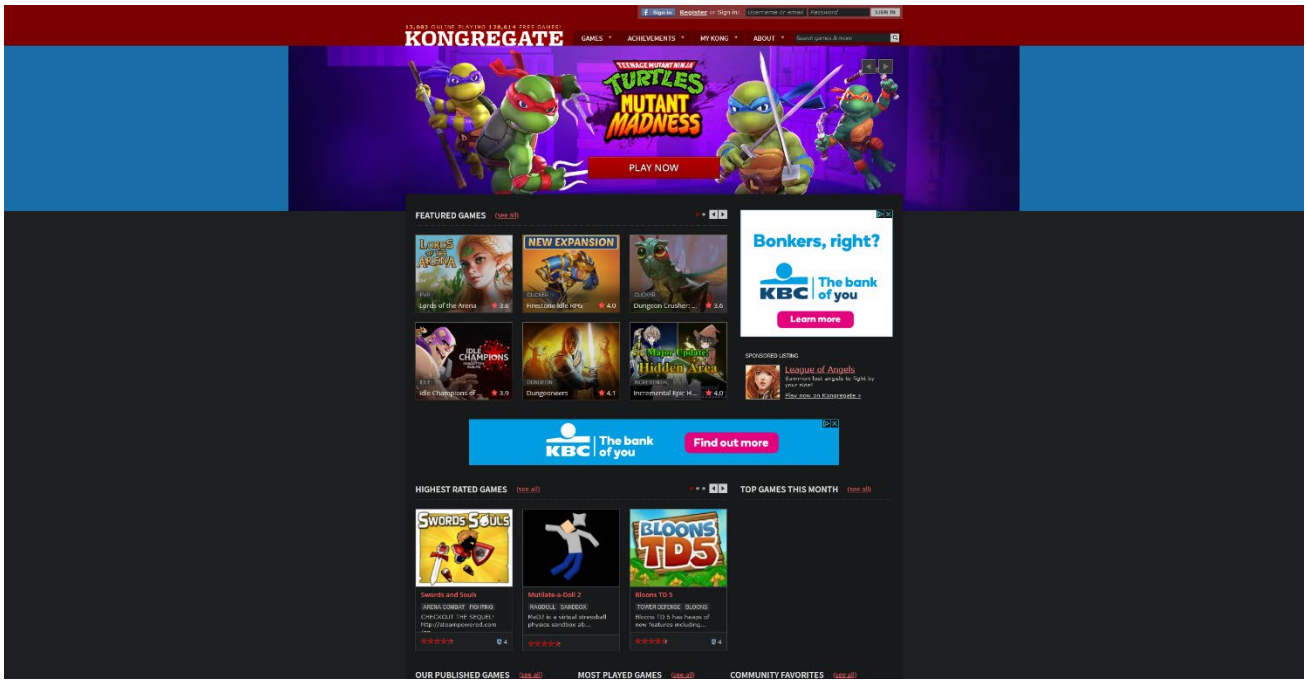
does not slow the loading speeds in most cases. Some websites are exceptional as you can see above. Easy blocker did not perform that well in RTE website and the reason for that is that the sending requests and getting the responses take time and what easy blocker does send the requests and blocks the response if it is coming from one of the domain lists else if its request that's going straight to the ad or tracking site, it comes back without a response but the load still counts as the request being sent and returned.

Its unfortunate but during the testing phase tmall.com a Chinese shopping site managed to go down and I could not get a consistent result from the site unfortunately.

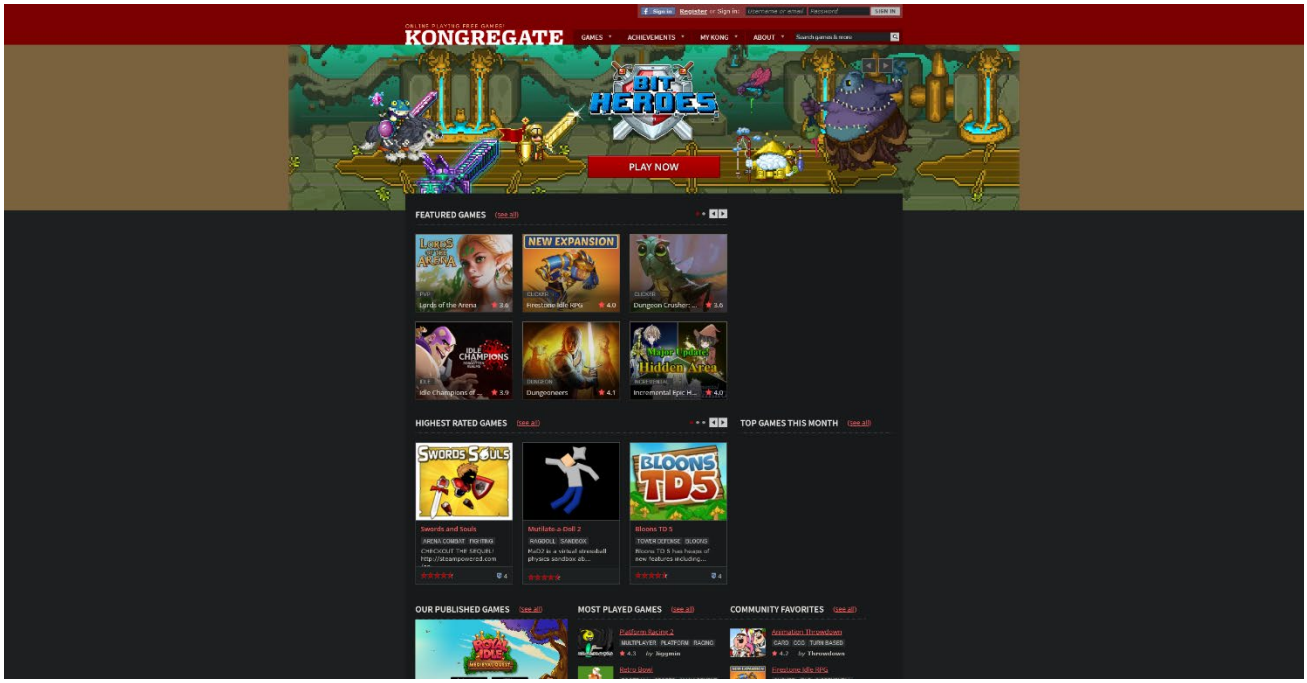
Visualisation Test:

The next test is to see how many ads it actually blocks from the website and how the view changes when using different technologies. In this case I am going to use popular website for games that kids would use. So, if you have a kid that use those websites and have ads it will be perfect for normal day use. I am using "Kongregate.com" website as a test environment.

This is how the website looks like when the user would use nothing against the ads.



This is how the website looks like when the user is using “uBlock Origins”.



As you can see there is already difference when ad block extension is running. There is not any KBC bank ads that are specifically targeting me. Who knows what they would put up for a kid just to get them to click the ad.?

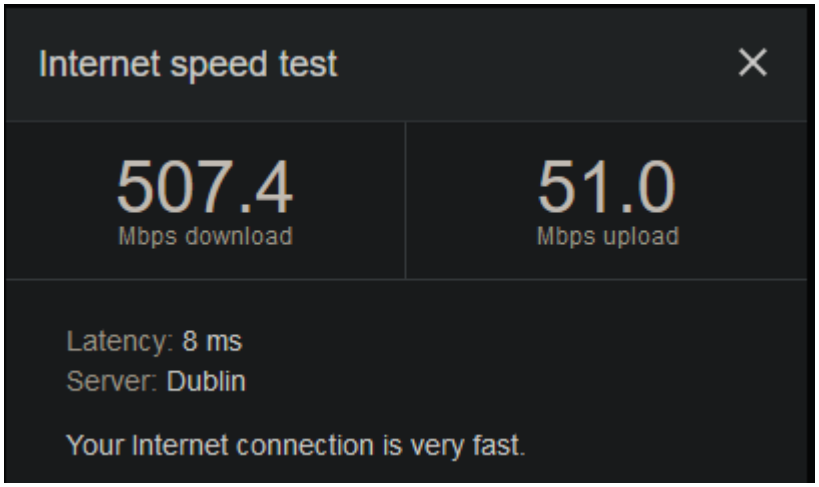
This is how the website looks like when the user would be using Easy Blocker.

As you can see the website still displays the elements of the ads, but they don’t appear as they are handled by Open-Wrt router blocking those domains away.

In conclusion to this testing not a bad choice to have an adblocker running even if it is a basic one as they will prevent malicious ads from being accessed. They completely remove the idea of ads and remove the position they are at. Yet it can only block it on one browser. This can become complicated as the user might need to look for another extension on another browser to have similar protection against ads.

While Easy blocker is not able to do the same as removing the position and changing up the website. What it provides is a block that run through your whole network instead and just blocks things around it. So, if a user does not want to tinker with it for too much, they can just enable it and it will work fine for most of devices that are using wired connection.

While Easy blocker myself I have not really noticed too many issues when trying to load a website or a place that does not allow adblockers preventing from seeing content. When domain list as big as 80k domains I found that not all the ads get blocked but maybe around 5% of the ads don't get blocked. This can change to the fact that there might be new domains for the ads, or the 3rd party list might not it filtered. This can be easily fixed by enabling bigger domain lists because the user is able to do so using Raspberry Pi. There hasn't been an issue where it slows down the browsing speed or anything. In fact, the download and upload speeds are the same which quite surprised me. As before I was working on a different technology to provide the same result, I encountered and issue that the internet speed gets slowed down dramatically and this was a big issue for the browsing experience, but for easy blocker this is not the case. In fact, we are paying for 500 Mbps and 50 Mbps upload and this is the result of running the speed test while Easy Blocker is enabled.

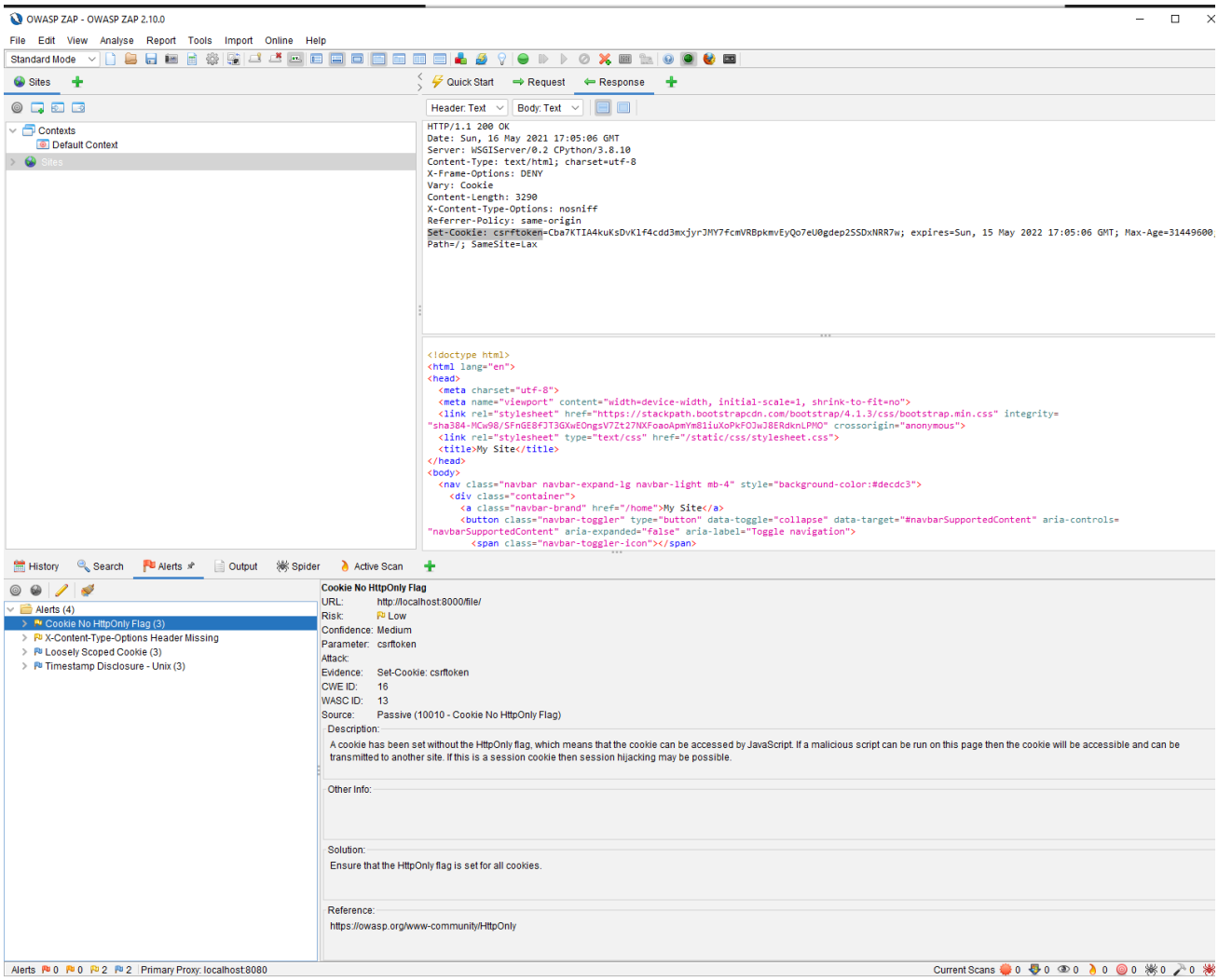


While on the other project I had issues with download speed going down as much as 40Mbps, which was extreme at that point.

The issue that I noticed while using this was when I was running isolated test and such. I had it connected to the internet switch beside my computer and when I turned Easy Blocker off. The internet would slow down, and things would take like 30 seconds extra to load for couple of minutes before everything goes back to normal. This issue is not major as long as Easy Blocker is running, its just the network adjusting to the route as the Easy Blocker is turned off that there is no more traffic going into it.

Security testing of API:

Next up is to test out API, I used “OWASP ZAP” to see if there are any major issues that the hackers could abuse. This program scans the website and looks for any major issues that hackers could find and abuse the system to the point that they could gain access. From the picture bellow you can see that the issues now are just HTTP flags and its nothing serious. As the API is being ran isolated it should be secure enough to not be targeted in any attack.



Integrity testing of API:

In this testing phase I used and abused my API for as much as I could to test out each individual features. This required me running feature by feature on the API, disabling some of the functions to see if everything is working without the other. Then how they respond when they are tested together and how do they respond during performance testing. I needed to test specifically admin access, just to know that its working as intended and that nobody else can just be an admin. This was to ensure role-based protection on what can each user do and how does API react when the there is a standard user going through upload process.

GUI Testing of API:

In this testing phase I ran test to see if all the functionality of buttons and pages work. That nothing is out of place and everything gets rendered how it should be. The database is getting display as it should be no matter what files are uploaded or what they are named or what information they hold. Checking each of the displayed components by disabling them and enabling them through the developers' options in browsers. This allows to check individual views and such without any constrains. I found several issues where certain pages were not picking up static stylesheet that I implemented, but now the issues have been fixed by just referring to a specific folder on 'base.html' page.

4.0 Conclusions

Describe the advantages/disadvantages, strengths, and limitations of the project

So, the advantages of this project are quite clear. Its an easy way to block ads as long as the system is setup correctly. It blocks most of the ads that are found on the websites, especially the banner ones, or the ones that popup. It allows smooth browsing experience without much of a hassle. It allows customising for users' own filters if needed or use huge compilation of various blocked domain lists.

If user need to whitelist something everything, user simply connects to the Open-Wrt router and he can whitelist or blacklist any domain that they want or not want to see, and the process is so seamless.

Compiling and making your own list could be hard to an extent but uploading it to work with router using API is very simple.

It increases the load times of websites but not by a lot, most people won't notice it but extra milliseconds are already a great progress for a full network coverage.

There is a main disadvantage that makes this project not as good as competitors like "Pi-hole". For my Raspberry Pi model which is 3B+, it does not have access point or in other words it does not work with wireless devices. Which limited this project effectiveness as the wireless devices do not get cover so, phones, laptops, TV, gaming consoles and other IoT devices that do not use ethernet cable are excluded from Adblocking service.

The strength of this project is probably that you can scale it up. Open-Wrt is an open-source project and users allowed to tinker with this technology for as much as they want. That means that this project is scalable and can be improved many in many ways by applying other packages and techniques to it.

The limitations would be that it depends on the device that people would try to implement this on. Not every single IoT device can become a router and not every router can run these types of firmware. Some devices will struggle with running huge compiles domain list as it would require something like raspberry pi to do so, even model depends as older models might not have the RAM capacity to enable this project to full extent.

5.0 Further Development or Research

With additional time and resources, which direction would this project take?

I think with more time and resources this project could definitely be improved in many ways.

First would be to enable artificial intelligence. I really wanted to accomplish this in my own time and have it ready for the deadline, but it was not possible to meet. I think having AI to filter the ads and compile the list itself would be an amazing feat. Using the data sets that are already online would definitely help as well as there are already sources for AI detecting malicious and click baiting ads or websites.

I think there would be a great idea to keep improving on the API as well to have a really nice communication between the client and the server. Allowing seamless changes without logging into second router to access the menus. This requires a lot of research and time to work on but there's documentation on how to export that data that is on the router to be able to change things within the API.

Improving API and how it handles everything could be done better especially the design of it. I did not put too much effort on how it looks like this API is not for public use but for private use only. I could definitely see improvement on the design, the way it looks and what it contains. Displaying the database is simple enough of a design there could be more complexity with AI element working providing lists and blocks in real time.

I really wanted to implement some visualisation, like graphs and statistics using "Graphana" or "Tableau". These can provide real time cloud statistics. I would use these for how much domains have it blocked, what's the constant speed of the internet. Add real time monitor on who is connected and using Easy Blocker.

I would definitely like to make this project for wireless too which I didn't manage to achieve because of my Raspberry Pi model. I would love to test it out and see how it works with wireless devices and how much it effects their speed and their load times in different applications and websites. The idea of being able to block ads on apps that keep showing them in your face.

I think this project has a lot potential outside adblocking alone. I will continue to work on this after its all done just so I could have a working and improved system that is unique in my house.

6.0 References

[1] DataProt. 2021. *A Not-So-Common Cold: Malware Statistics in 2021 | DataProt*. [online] Available at: <https://dataprot.net/statistics/malware-statistics/> [Accessed 4 May 2021].

[2] GitHub. 2021. *gorhill/uBlock*. [online] Available at: <https://github.com/gorhill/uBlock>.

[3] Brown, R., 2021. *Welcome to the OpenWrt Project*. [online] OpenWrt Wiki. Available at: <https://openwrt.org/> [Accessed 16 May 2021].

Pi-hole.net. 2021. *Home*. [online] Available at: <https://pi-hole.net/> [Accessed 16 May 2021].

GitHub. 2021. *openwrt/luci*. [online] Available at: <https://github.com/openwrt/luci/wiki> [Accessed 16 May 2021].

Python.org. 2021. *Welcome to Python.org*. [online] Available at: <https://www.python.org/> [Accessed 16 May 2021].

Djangoproject.com. 2021. *The Web framework for perfectionists with deadlines | Django*. [online] Available at: <https://www.djangoproject.com/> [Accessed 16 May 2021].

Zaproxy.org. 2021. *The ZAP Homepage*. [online] Available at: <https://www.zaproxy.org/> [Accessed 16 May 2021].

2021. [online] Available at: <https://www.cloudflare.com/learning/dns/what-is-dns/> [Accessed 16 May 2021].

MuleSoft. 2021. *What is an API? (Application Programming Interface) | MuleSoft*. [online] Available at: <https://www.mulesoft.com/resources/api/what-is-an-api> [Accessed 16 May 2021].

7.0 Appendices

This section should contain information that is supplementary to the main body of the report.

7.1. Project Plan

		Project	**149 days?**	**Mon 28/09/20**	**Thu 22/04/21**	
		Research	**10 days**	**Mon 28/09/20**	**Fri 09/10/20**	
		Determine project scope	8 hrs	Mon 28/09/20	Mon 28/09/20	
		Look into competitors	1 day	Tue 29/09/20	Tue 29/09/20	
		Look into python	1 day	Wed 30/09/20	Wed 30/09/20	
		Research machine learning	2 days	Thu 01/10/20	Fri 02/10/20	
		Project pitch video	**6 days**	**Mon 12/10/20**	**Sun 18/10/20**	
		Prepare script	2 days	Mon 12/10/20	Tue 13/10/20	
		Practice	2 days	Wed 14/10/20	Thu 15/10/20	
		Analysis comple	0 days	Thu 15/10/20	Thu 15/10/20	
		Project proposal and ethics doc	**11 days?**	**Mon 26/10/20**	**Sun 08/11/20**	
		Objectives	1 day	Mon 26/10/20	Mon 26/10/20	
		Background	1 day	Tue 27/10/20	Tue 27/10/20	
		Technical approach	1 day	Wed 28/10/20	Wed 28/10/20	
		Project Plan	2 days	Thu 29/10/20	Fri 30/10/20	
		Technical Details	1 day	Mon 02/11/20	Mon 02/11/20	
		Evaluation	1 day	Tue 03/11/20	Tue 03/11/20	
		Invention Disclosure Form	2 days	Wed 04/11/20	Thu 05/11/20	
		Requirements	**14 days**	**Mon 12/10/20**	**Thu 29/10/20**	
		Gather requirements	7 days	Mon 12/10/20	Tue 20/10/20	
		Research	7 days	Mon 12/10/20	Tue 20/10/20	
		Mid point implementation	**3 days**	**Mon 09/11/20**	**Wed 11/11/20**	
		Documentation	3 days	Mon 09/11/20	Wed 11/11/20	
		Video presentation	3 days	Mon 09/11/20	Wed 11/11/20	
		Monthly journey	**146 days**	**Thu 01/10/20**	**Thu 22/04/21**	
		October	1 day	Thu 01/10/20	Fri 06/11/20	
		November	1.6 days	Fri 06/11/20	Mon 09/11/20	

7.2. Reflective Journals

REFLECTIVE JOURNAL

October

Mindaugas Prismantas x17489412

I started October off quite nervous as I still was not sure about my final year project idea. I had to go online around to make some research see what I can do specialising into cyber security. I talked with some of my friends that are in my course but doing different specialisation, on how they are looking for information and researching about the project.

I have learned that I did not know as much as I thought I did. This was not really a surprise, but I needed to put in more time and effort to get things started with my final year project. I prepared the pitch video and I have started doing the ethics form and a bit of proposal even if I did not get my project idea approved, I wanted to get ahead of my work.

Once I got my project approved, I jumped straight ahead to complete the project proposal document specifying what I am doing my project and how I am approaching it. I contacted my supervisor, and he gave me some guidance on how I should approach and just start it off. He told me to do as much research as I can. So that is what I'm going to do before start project.

REFLECTIVE JOURNAL

November

Mindaugas Prismantas x17489412

November was good, and things seemed nice until I started to fall behind my work, and I couldn't make much progress on my final year project. I learned for this month that if I do not start my assignments as soon as possible it will hurt all my time scheduling around final year project. I became more and more stressed as I met deadline after deadline it was hard to focus just on project and try to achieve a good grade on other modules CAs. I postponed my work on final year project so I can catch up on all the deadlines and by the end of November I should be in a good place where I can make loads of progress onto my project.

I understood that working on the CAs and final year project is not going to be a simple task so I need to prepare a time schedule on how much time I should work on things each day as the plan will keep me on the right track and help me focus on the most important issues at first.

I feel like it’s a good thing that I managed to be overwhelmed with work so early so I get to learn that nothing can be done if things are on hold or brushed off.

REFLECTIVE JOURNAL

December

Mindaugas Prismantas x17489412

Starting with December it was a scary month as when I was trying to get progress on my project, I was getting stuck, and it seemed like I could not get any of my ideas off to start the project and progress towards it. I was rethinking the ideas and thinking that this project idea might have been a bad one. After meeting with my supervisor, he gave me an idea how I should get all my things onto a list and then try my best to tackle one issue at the time. I needed rethink the steps and slowly start progressing. He told me that focusing on presentation that was coming up would help me to visualise the project and help to understand the actual requirements and what I want to achieve. This was not an easy job especially when I was already behind work but slowly and surely, I managed to make a breakthrough.

Taking notes and slowly ticking the tasks away helped me stay focus on one path and helped me not to be discouraged with my project idea as I started to doubt myself and thinking that it just would not work.

It a great learning experience that I get to go through and understand that nothing can be done if I am trying to accomplish something without a plan. Planning ahead is a good way to stay focused.

REFLECTIVE JOURNAL

January

Mindaugas Prismantas x17489412

January was one of the slower months as I was only getting into my project. Getting piece by piece together. Trying out one of the ways which was using just Windows to get ads blocking, managed to make my pcs internet break as I did not know “host” file cannot be bigger than 1Mb or it will take long while to parse it before the computer can connect to the internet. This issue was a funny one and I was not sure what was going on. I fixed the issue and understood that this way I will not be able to make any progress and I needed to look for something else. I was trying to make a python script that can block DNS that I did not allow. This took a while and it still was not successful as I thought it would be. I was going back and forward with different approaches that is why I felt that the month was going a bit slow. As I was stuck on a blockage on not knowing which way I should go.

Even with a plan I was still getting discouraged every time my code or something else was not working. Its hard to explain but it was not a nice feeling to know any effort I did was just for nothing as it was not working and with my personality when things do not work, I just scrap the idea and move onto the next thing instead of trying to improve or elevate the issue and look another way to complete it.

I started to take notes of what I do wrong and just see what helps me to progress. I looked into YouTube videos to see how to keep up with projects when you’re stuck and see what I need to do to improve my scrapping ideas trait as its definitely not a way to complete a project.

REFLECTIVE JOURNAL

February

Mindaugas Prismantas x17489412

February was probably one of my successful months, I was already catching up onto my project work and I managed to make progress. I had meetings with my supervisor that was asking questions and trying to understand my project path. This helped me tremendously as it helped me to understand that I am doing the right thing. He gave me few suggestions in a way to use raspberry pi which I already had and tried making progress, but I was not good at it. I made a massive breakthrough with it; I learned a lot about how network works, and it made me learn how to quickly navigate through Linux OS. It helped me a lot in my penetration labs too. I was happy how much progress I was making, changing my raspberry pi into a router, and understanding how routers work. It was great fun to see project getting somewhere.

I learned that sometimes getting stuck on things is not a bad thing especially as you keep trying to make progress and learn something new. When this project started to move, I was more motivated than ever to work on it and just keep building on to it.

Motivation was a big factor on working on the project especially when the progress is going nowhere. It is hard to get yourself motivated and work on it when everything you do is not working. When you make a breakthrough, and you are able to keep going it feels really nice and just made me work much more on the project.

REFLECTIVE JOURNAL

March

Mindaugas Prismantas x17489412

March was a difficult month as it was starting to get continues assessments and project work. I thought I was going to struggle but I already experienced what happened in December when I was falling behind. I started to work on my CAs straight away, it was not a big issue to get some project work done. I was onto of my work. I skipped few days there and there to get some CA work done but that did not make me fall behind whatsoever. It felt great knowing that I did not need to worry about two or more issues at the same time. The stress levels where low and it helped me to keep going with college work. Yet I started to feel pressure that the deadline for project was getting closer and closer which made me to start panicking that I don't have much time left. I started to think more and more about the deadlines instead of the work ahead which definitely made me feel a bit worse.

It was a good feeling knowing that I am not repeating the same mistakes I did last months and getting my things together definitely helped me out with project progress. Little by little though stress was creeping up and making me more pressured knowing that time is ticking, and I am getting closer to the deadline.

REFLECTIVE JOURNAL

April

Mindaugas Prismantas x17489412

April was an okay month, we had couple of issues with college having Moodle down, not able to access notes and or college library definitely slowed of my college work down. It did not make me panic but it added additional stress from knowing that deadlines are coming closer, and I have little time to get my work done. I started to struggle with project, I was not making much progress apart from touching some of the parts to make it more accessible and more secure. With more stress and less progress, it was very demotivating to keep working on it. I did not give up, but it was just a big struggle that I did not want to do anymore. I am getting the ideas that my project just was not good enough and that everyone else was doing really well. Negative thoughts went through my head and slowed down my progress more. It was not the best thing for me but having few chats with my supervisor kept me motivated and helped me to make some of the progress that I think I alone would not have done if I did not get encouraged. My supervisor told me to keep going by the marking scheme and see where I can keep getting better marks and what I should keep looking out for. Just this little push kept me going without worry too much that my project might not be the best suited for me.

Just having issues with project demotivated me loads yet I kept going even if it was a bit of progress, I did not care. I wanted to do as much as I can and do whatever I can just so I would not regret this. I kept telling myself that I need to keep going and it's the last year of college just another one-month push and I'm done. So, I kept going and I planning on going on until its done.

I really must thank my supervisor as he knew that sometimes things like that happen that people can struggle with their projects. Just him saying that I was getting good marks in one section and I need to improve in the other kept me going to get better grade for this project and that its not that bad of a project from what I have already.