Multi-UI for Multi-Tenant/Multi-Domain Application

Aravind Kumar R, Dr. K Thangaraj

[1]B. Tech IT Final Year Student, Graduate, Dept. of IT, Sona College of Technology, Salem, Tamil Nadu, India
[2]Assistant Professor, Dept. of IT, Sona College of Technology, Salem, Tamil Nadu, India

---***---

Abstract - *This project aims to give the client a seamless and best User Experience (UX) while using our software product called "Traffic Lenz". Traffic Lenz is Multi-tenant, Multi-domain Web-based Traffic Data Analysis & Management software, which already has a default User Interface. Traffic Lenz is used by many clients in an OEM mode with the client's own branding and domain. Our goal is to improvise and modernize the User Interface of Traffic Lenz without losing any of the functionality that Traffic Lenz has been doing already. We would also like to ensure that each of the OEM serviced clients has a unique UI/UX to choose and use. In the end, the clients of Traffic Lenz will be given at least 6 UIs out of which they will select one that they like. The designing of the User Interfaces will be done in HTML, CSS and JavaScript by making use of the Materialize Theme provided by Google. After the UIs are designed, it will be integrated into Traffic Lenz which is developed in Django - A Web-Framework in Python. So, by changing the UI of Traffic Lenz, the clients will have more choice to customize the portal as per their needs to showcase to their customers.*

Key Words: Multi-UI, Multi-Domain, Multi-Tenant, User Interface, User Experience, UI/UX Design, Django, Python, HTML, CSS, JavaScript

1. INTRODUCTION

The application called "Traffic Lenz" is a Web-Based application that is used for Traffic Analysis and Management for more than 2 years till now. Traffic Lenz is used by many clients all around the globe in OEM mode such that the clients use the application with their own branding and domain. This web application has been developed and maintained by a company called Datacorp Traffic Private Limited which is located in Bangalore. This web application has been in usage for around 2 years and this application is used in many client domains. This web application already has a default User Interface that was also developed two years ago and is still in usage. Now, our goal is to create and integrate Multiple User Interfaces and integrate it with Traffic Lenz such that the User Interface of our web application changes with respect to the domains of the clients. That is, earlier all of the clients used the same User Interface of Traffic Lenz in their domain and now after the process of integration of Multi-UI is over, the User Interface rendered in the client's side of Traffic Lenz will be different with one another. This is done so that the clients can have seamless and best User Experience while using Traffic Lenz

2. METHODOLOGY

In order to make the project working, we are going to make use of the concept of models in Django. Models in Django is nothing but databases connected with the web application.

Fig -1: Django Logo

Django is a Web Framework that is built with python. Django makes the developer develop the web applications in a faster way. It by default has the options for creating a login page. So, we do not need to worry about the creation of the login page by ourselves. It also provides the necessary security needed to maintain the login pages, like storing the password in the encrypted format without storing them in plain text. Also, the website administrator will not be able to view the password of the users of that website. Hence Django provides almost everything needed to build any kind of website. Because of that, you do not need to create the modules from scratch by yourself. Django will take care of the hassle web development process. [1]

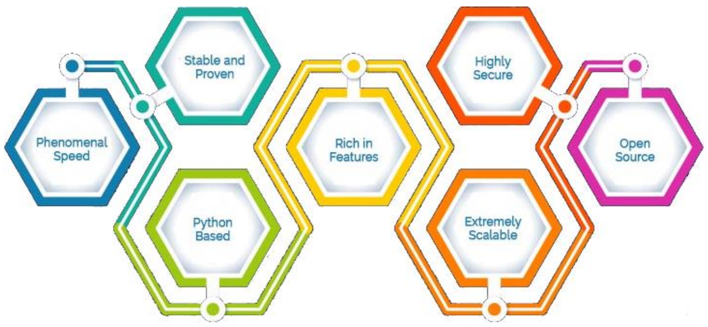
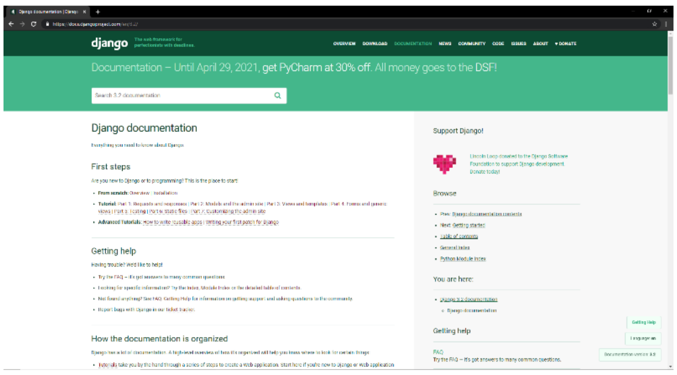


Fig -2: Features of Django illustrated

In today's modern world, Machine Learning and Artificial Intelligence play a very important role. Today we have Self-Driving cars, Robots doing surgery in hospitals, etc. which all are built and powered with Artificial Intelligence. In the field of web development, we have some AIs that can identify and delete posts on social media that may disturb world peace. Artificial Intelligence systems cannot function without the concepts of Machine Learning. So basically, Machine

Learning and Artificial Intelligence have already become an inevitable part of the human race. Today most of the Machine Learning and Artificial Intelligence modules are built over Python and R programming languages. Since the Django Web-Framework is built with Python, the web applications built with Django can also be easily integrated and built with the Machine Learning modules, unlike other backend web-development tools which call for the ML module only if needed. So, Django provides much feasibility with Machine Learning applications.

Django was developed in Japan around 2003-2005 by a team of full-stack web developers who worked in a Newspaper company. They were responsible for creating and maintaining the company's website. So, after creating lots of websites, the team found out the common pattern and design code among websites. So, the team decided to build the web framework for a faster web-development process. Later in July 2005, the Django framework was made open-source by its developers. Later a lot of bug fixes were made in Django and today, Django is free for anyone to use. It has an active community, well-explained documentation and moreover, Django has thousands of developers maintaining and updating this open-source project. [2]


Django documentation | Django
https://docs.djangoproject.com/en/3.2/
django The web framework for perfectionists with deadlines.
OVERVIEW DOWNLOAD DOCUMENTATION NEWS COMMUNITY CODE ISSUES ABOUT ♥ DONATE
Documentation – Until April 29, 2021, get PyCharm at 30% off. All money goes to the DSF!
Search 3.2 documentation

Django documentation

Everything you need to know about Django.

First steps

Are you new to Django or to programming? This is the place to start!

- From scratch: Overview | Installation
- Tutorial: Part 1: Requests and responses | Part 2: Models and the admin site | Part 3: Views and templates | Part 4: Forms and generic views | Part 5: Testing | Part 6: Static files | Part 7: Customizing the admin site
- Advanced Tutorials: How to write reusable apps | Writing your first patch for Django

Getting help

Having trouble? We'd like to help!

- Try the FAQ – it's got answers to many common questions.
- Looking for specific information? Try the Index, Module Index or the detailed table of contents.
- Not found anything? See FAQ: Getting Help for information on getting support and asking questions to the community.
- Report bugs with Django in our ticket tracker.

How the documentation is organized

Django has a lot of documentation. A high-level overview of how it's organized will help you know where to look for certain things:

- Tutorials take you by the hand through a series of steps to create a Web application. Start here if you're new to Django or Web application

Support Django!

Lincoln Loop donated to the Django Software Foundation to support Django development. Donate today!

Browse

- Prev: Django documentation contents
- Next: Getting started
- Table of contents
- General Index
- Python Module Index

You are here:

- Django 3.2 documentation
  - Django documentation

Getting help

FAQ
Try the FAQ — it's got answers to many common questions.

Getting Help
Language: en
Documentation version: 3.2

Fig -3: Django Documentation in Django's official website

3. LITERATURE SURVEY

3.1 Designing User Interfaces

In this paper, we are going to discuss how we developed 4-5 User Interfaces for the Traffic Lenz application. A designer designing a User Interface is similar to an artist doing a painting. This is because User Interface is all about the experience that the user is going to get while using the application, or in short, it is all about User Experience (UX). So, we cannot directly jump into coding the user interface out. We have to draw and see the user interfaces in some kind of rough format, like an artist drawing a rough sketch on a rough paper, before proceeding with the actual canvas painting. For designing a User Interface, we can use paper, but that will not serve the purpose. The UI is going to be used on a computer or a mobile phone. So, we have to use the computer itself to design the rough UI. Many use Adobe Photoshop to design a User Interface. But Adobe Photoshop is not a tool that is dedicated to User Interface designs. There are dedicated tools for User Interface designing like Figma, Adobe XD, Pencil, etc. We are using Adobe XD to draw a rough UI before we proceed to develop an actual working UI.

We design this rough User Interface before we start developing the actual UI because designing a rough UI takes very little time. So, for any UI changes, we can easily do the change and see by ourselves how the change feels like. If we are coding out the User Interface and if there is a situation to change something in UI, the process of coding will take more time for implementation, than drawing and seeing the change in UI designing tools. After the rough sketch is complete, the front-end designer works on the implementation of the actual User Interface in HTML, CSS and JavaScript by having the rough sketch as a reference. Today, the UI designing tools also comes with options to animate elements, going to other pages, etc. So, we can also get the rough User Experience with the designing tools before we move to code the UI.

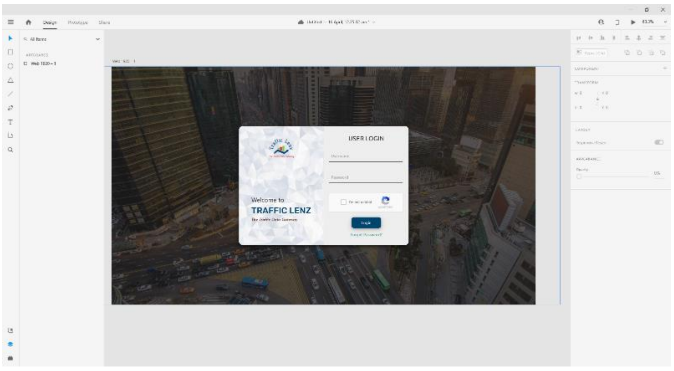


Fig -4: Designing a Login Page of Traffic Lenz in Adobe XD

3.2 Implementation of working User Interfaces

Traffic Lenz is a Web-Based application that consists of a Dashboard, Tables, Forms and some Utility Tools. The dashboard contains some Graphical Representation of data in form of Pie-Charts and Bar Graphs. The tables are used to show the data fetched from the Database. The forms are used to create new users and to get small inputs from the users. So, to design and develop the User Interface for Traffic Lenz, we have used HTML, CSS and JavaScript. The CSS styles for each of the theme is written in an external CSS file and these external CSS files are linked with HTML pages. These CSS styles use the HTML IDs and Classes to specify the styles and designs for a particular element in the desired way.

The CSS specifies the font style, colour, font size, background colour, table styles, form styles, etc. of the HTML elements. Likewise, JavaScript is used to specify the animation effects for the HTML elements so that the User Experience (UX) is good and engaging. The JavaScript for various themes is also written in external JS files and these files are linked to HTML in the same way as we linked CSS files with HTML. On the whole, the HTML, CSS and JavaScript files are properly arranged and organized inside folders and the CSS and JavaScript files are properly linked with HTML to get the result as the User Interface designed with Adobe XD.

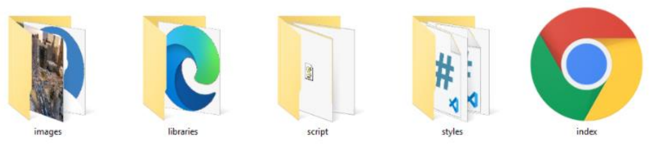


Fig -5: HTML, CSS and JavaScript files associated with login page of one of the UIs

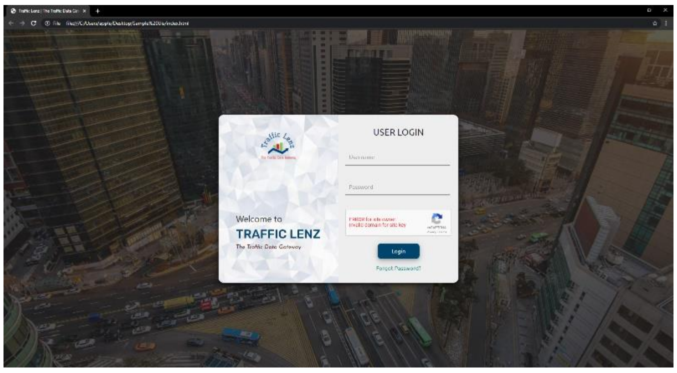


Fig -6: Developed Front-end UI shown in browser

3.3 Process of Integration with Traffic Lenz

The Traffic Lens Application was developed two years before. So, it has been using a default User Interface. So, while adding more User Interface to the application, CSS and JS files for the already existing default User Interface is not disturbed. Whereas the CSS and JavaScript files for the newly added UIs are properly separated inside different folders. In other words, every new UI has a separate folder and the CSS and JavaScript files needed for that UI are organized inside that folder. If a UI is needed, the CSS and JavaScript files of that particular UI is linked with the HTML. This is done by Django - a python backend Web-Framework.

The backend of Traffic Lenz is written with the help of the Django Framework. This backend of the application decides which User Interface to be rendered and links the CSS and JavaScript files needed for that UI with the HTML files. So, when a client uses the application, a particular User-Interface is rendered and the client gets a seamless user experience while using that UI.

3.4 Making the User Interface Responsive such that it can be used in all the devices.

Usually, a Web-Based application will be hosted in a domain and whenever someone enters the URL of that application in a browser, the webpage will be generated and loaded on the user's device. But in today's world, users can access any website through Desktops, Laptops, Smartphones, Tablets or even through Smartwatches. There are also devices like Smart TV, Smart Fridge, Smart Displays in Cars, etc. that also provide users with the internet and browser facilities. So today, any given website or web-based application should be able to adapt and change itself as per the resolution of the screen it is being accessed.

As explained earlier, the Traffic Lenz application consists of a Dashboard, Tables and Forms which challenges us to make the application to be responsive to the device being used by the client. The already existing default User Interface has been developed to be responsive, but we had to develop the newly added User Interfaces also to be responsive with the client's device. The responsiveness features include hiding the navigation menu to a sidebar, showing the most important data in tables and hiding the least important information inside a dropdown, etc. when the screen size is not as big as needed. This is done with the help of breakpoints in CSS and JavaScript. Hence, our clients will now get a seamless user experience no matter whatever device they use to access Traffic Lenz.

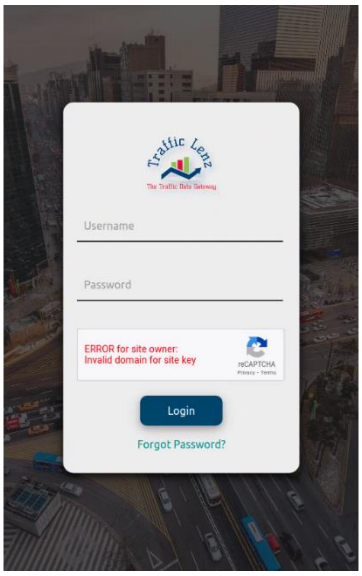


Fig -7: Login Page of Traffic Lenz being responsive in mobile view

3.5 Making the User Interface to be rendered based on domain.

The Traffic Lenz application has been hosted on multiple domains of our clients. So, for the clients to get a seamless experience of a particular UI, we have to render the entire Traffic Lenz in a particular User Interface. That is, when a UI is being rendered, it should display all the HTML elements customized for the UI and the customization should not be partial or incomplete. This paper is all about Domain-based User Interface selection. So, our application holds all the necessary code to render all of the UI. But whenever a user tries to access Traffic Lenz via any of the hosted domain, Traffic Lenz must be rendered with the User Interface associated with that particular domain. This is done in the backend.

The Django part of our application holds a record of which UI is associated with which domain in a model called "Domain Resources". Models in Django are nothing but databases. This Domain Resource model has all the needed fields to store the details of every client's domain. The database has fields to store the client's name, the client's location, the client's logo, client's domain along with a field to store which theme is to be rendered with which client's domain. The record which stores the theme from the field is fetched and is used inside the HTML code so that the associated UI can be rendered. By placing if checks for UIs in the HTML files, we can easily change the IDs and Classes associated with each of the HTML elements so that an element with the required IDs and Classes gets rendered in that particular theme. Hence when a user uses Traffic Lenz through any of the client's domain, the UI is fetched from the database and the HTML elements are generated accordingly. So, the user will be able to get a seamless experience with the theme they are using Traffic Lenz.

4. EXISTING SYSTEM

The already existing system of Traffic Lenz uses the same User Interface for all the domains. The default user interface has been built with Materialize UI theme. But based on different domains and clients, the background image of the page is changed, the colour scheme of the theme is altered, the logo of the page is changed and likewise some minor changes are implemented. On the whole, most of the User Interface remains the same between all the domains. But since most of the UI remains the same, the clients do not get a good and vivid user experience with the already existing interface.

5. PROPOSED SYSTEM

After the integration has been done, the User Interface gets changed completely based on the domain. The clients will now get a seamless experience with the newly integrates User Interface. The new integrated User Interface gives the users of Traffic Lenz a more clear and detailed impressions about the data shown. So, when a user uses the Traffic Lenz application from two different clients, he will now be able to experience the application in two different versions. This is because each client will be hosting Traffic Lenz in their domain and the UI gets changed based on the domain. Earlier, when there was a common UI for all domains, some users who used Traffic Lenz from two or more clients did not get a vivid experience. As the UI has been integrated, those users will now get a vivid experience and a different version of Traffic Lenz from different clients.

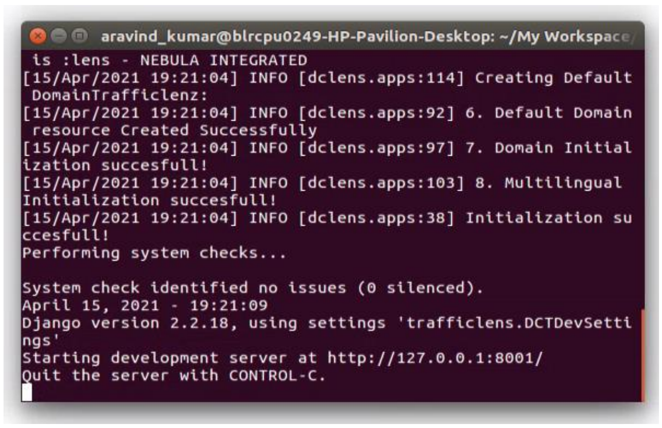
6. IMPLEMENTATION

When a theme was developed from scratch, it was developed in a model Webpage that looks like Traffic Lenz. It was not developed over the live Traffic Lenz application. No backend was connected while the themes were developed. The themes were developed with HTML, CSS and JavaScript. After the model Webpage had been completely developed, the HTML files were not needed anymore, while the CSS and JS files were stored and organized inside a folder. There are different folders for different UIs with the corresponding CSS and JavaScript files organized within them.

After the CSS and JavaScript files were stored, the backend of Traffic Lenz was programmed in such a way that when a given UI is needed to be rendered, the corresponding CSS and JS files are imported and linked with the HTML pages. Also, for some HTML elements that remain exclusive to a particular theme, those elements are added by placing them inside if checks and checking for the theme that is rendered. Such if checks are also used to change the IDs and Classes of the HTML tags and elements so that all the elements get customized as per the UI in which the page is getting rendered.

The backend of Traffic Lenz is written with Django - a python-based web framework. Django part of the application has been programmed to keep a record of UIs associated with the domains. So, when a Traffic Lenz is accessed by a user through a client's domain, Django checks the domain and renders Traffic Lenz in the User Interface associated with the domain. When the HTML part of code differs in some places, the if checks placed inside the HTML checks for the UI to be rendered and decides whether to render that part of HTML or not.

Hence with different domains of Traffic Lenz, different User Interface is getting rendered.



```
aravind_kumar@blrcpu0249-HP-Pavilion-Desktop: ~/My Workspace
 is :lens - NEBULA INTEGRATED
[15/Apr/2021 19:21:04] INFO [dclens.apps:114] Creating Default
 DomainTrafficlenz:
[15/Apr/2021 19:21:04] INFO [dclens.apps:92] 6. Default Domain
 resource Created Successfully
[15/Apr/2021 19:21:04] INFO [dclens.apps:97] 7. Domain Initial
ization succesfull!
[15/Apr/2021 19:21:04] INFO [dclens.apps:103] 8. Multilingual
Initialization succesfull!
[15/Apr/2021 19:21:04] INFO [dclens.apps:38] Initialization su
ccesfull!
Performing system checks...

System check identified no issues (0 silenced).
April 15, 2021 - 19:21:09
Django version 2.2.18, using settings 'trafficlens.DCTDevSetti
ngs'
Starting development server at http://127.0.0.1:8001/
Quit the server with CONTROL-C.
```

Fig -8: Traffic Lenz hosted locally over port 8001

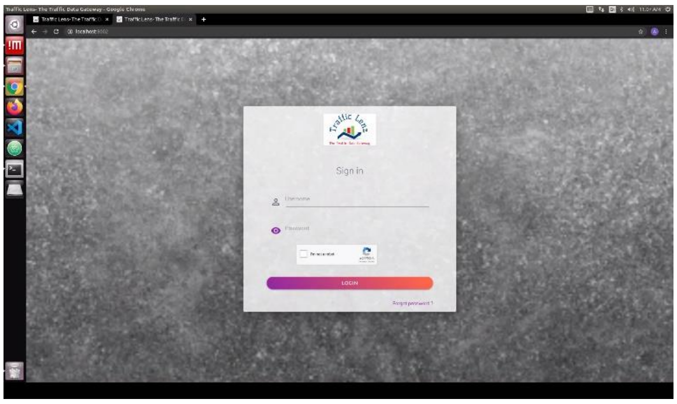


Fig -11: Login Page of Traffic Lenz is hosted over port 8002 and is rendered in Theme 2 (default theme)

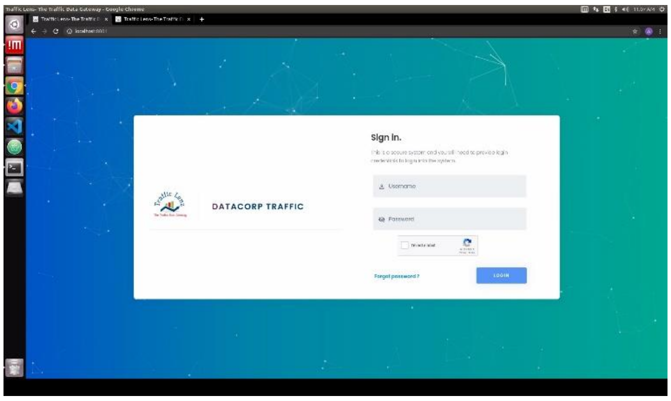
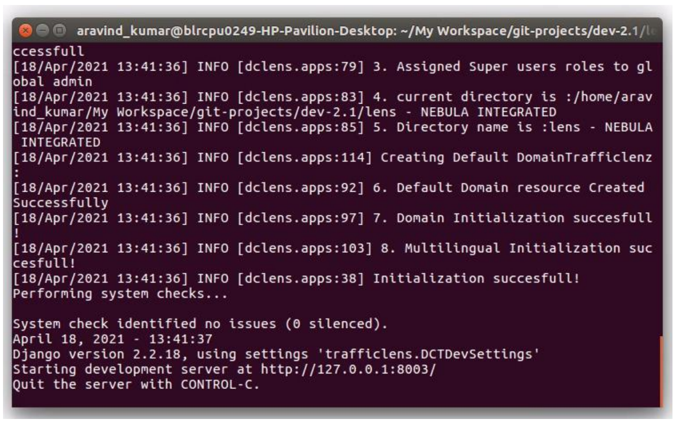


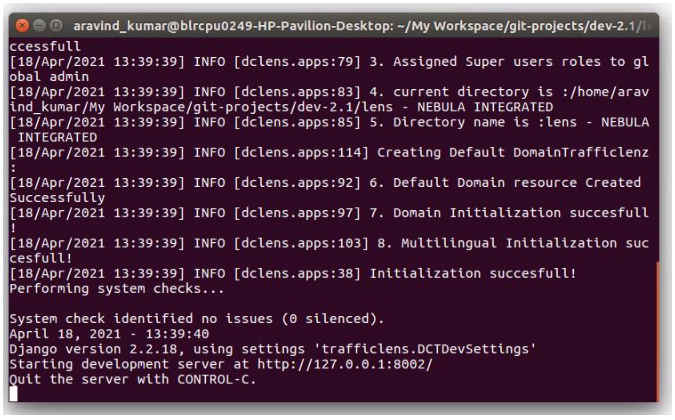
Fig -9: Login Page of Traffic Lenz is hosted over port 8001 and is rendered in Theme 1



```
aravind_kumar@blrcpu0249-HP-Pavilion-Desktop: ~/My Workspace/git-projects/dev-2.1/
ccessfull
[18/Apr/2021 13:41:36] INFO [dclens.apps:79] 3. Assigned Super users roles to gl
obal admin
[18/Apr/2021 13:41:36] INFO [dclens.apps:83] 4. current directory is :/home/arav
ind_kumar/My Workspace/git-projects/dev-2.1/lens - NEBULA INTEGRATED
[18/Apr/2021 13:41:36] INFO [dclens.apps:85] 5. Directory name is :lens - NEBULA
 INTEGRATED
[18/Apr/2021 13:41:36] INFO [dclens.apps:114] Creating Default DomainTrafficlenz
:
[18/Apr/2021 13:41:36] INFO [dclens.apps:92] 6. Default Domain resource Created
Successfully
[18/Apr/2021 13:41:36] INFO [dclens.apps:97] 7. Domain Initialization succesfull
!
[18/Apr/2021 13:41:36] INFO [dclens.apps:103] 8. Multilingual Initialization suc
cesfull!
[18/Apr/2021 13:41:36] INFO [dclens.apps:38] Initialization succesfull!
Performing system checks...

System check identified no issues (0 silenced).
April 18, 2021 - 13:41:37
Django version 2.2.18, using settings 'trafficlens.DCTDevSettings'
Starting development server at http://127.0.0.1:8003/
Quit the server with CONTROL-C.
```

Fig -12: Traffic Lenz hosted locally over port 8003



```
aravind_kumar@blrcpu0249-HP-Pavilion-Desktop: ~/My Workspace/git-projects/dev-2.1/
ccessfull
[18/Apr/2021 13:39:39] INFO [dclens.apps:79] 3. Assigned Super users roles to gl
obal admin
[18/Apr/2021 13:39:39] INFO [dclens.apps:83] 4. current directory is :/home/arav
ind_kumar/My Workspace/git-projects/dev-2.1/lens - NEBULA INTEGRATED
[18/Apr/2021 13:39:39] INFO [dclens.apps:85] 5. Directory name is :lens - NEBULA
 INTEGRATED
[18/Apr/2021 13:39:39] INFO [dclens.apps:114] Creating Default DomainTrafficlenz
:
[18/Apr/2021 13:39:39] INFO [dclens.apps:92] 6. Default Domain resource Created
Successfully
[18/Apr/2021 13:39:39] INFO [dclens.apps:97] 7. Domain Initialization succesfull
!
[18/Apr/2021 13:39:39] INFO [dclens.apps:103] 8. Multilingual Initialization suc
cesfull!
[18/Apr/2021 13:39:39] INFO [dclens.apps:38] Initialization succesfull!
Performing system checks...

System check identified no issues (0 silenced).
April 18, 2021 - 13:39:40
Django version 2.2.18, using settings 'trafficlens.DCTDevSettings'
Starting development server at http://127.0.0.1:8002/
Quit the server with CONTROL-C.
```

Fig -10: Traffic Lenz hosted locally over port 8002

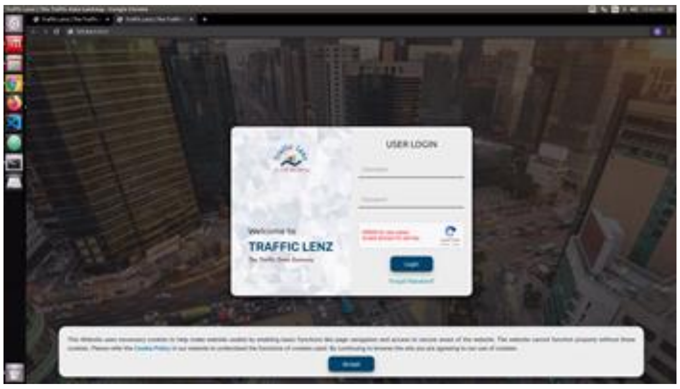


Fig -11: Login Page of Traffic Lenz is hosted over port 8003 and is rendered in Theme 3

7. CONCLUSION

The project of integrating Multiple UIs with Traffic Lens and rendering them for each domain has been successful. In total, about 6-7 themes are integrated with the Traffic Lenz application. Now the users of the Traffic Lenz web application will be able to view the theme in various different themes. Based on the domain in which Traffic Lenz is accessed, the web application now automatically renders the configured theme associated with that domain. So the project is successful in giving the clients a seamless experience and the users a vivid and detailed impression of the data that they see. In future, if needed we will design few more themes and integrate them with Traffic Lenz and make them get rendered based on the domain accessed in the same way.

REFERENCES

[1] Mozilla Developer's Network https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction

[2] Django Documentation https://docs.djangoproject.com/en/3.2/

AUTHORS

First Author – Aravind Kumar R, B. Tech. IT Final year, Sona College of Technology, aravind320291@gmail.com.

Second Author – Dr. K. Thangaraj, Assistant Professor, Sona College of Technology, thangarajkesavan@gmail.com.