**Acknowledgement**

I want to express my gratitude towards my mentor **Mr. Manish Kumar Gupta** to provide me such a great opportunity to do project on the topic **URL Shortening Service** at **Aranya consulting pvt. Ltd., Delhi**. I am thankful to him for creating such a wonderful environment during the period of Internship.

I also want to thank Dr. Rajeev Trehan, Head of Training and Placement office, and Dr. Geeta Sikka, Head of department (CSE), NIT Jalandhar who provided me with the opportunity to undergo this Summer Internship.

I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

**About the Company**

**ARANYA CONSULTING**

Aranya Consulting Private Limited is a Private incorporated on 22 February 2011. It is classified as Non-govt Company and is registered at Registrar of Companies, Delhi. Its authorized share capital is Rs. 200,000 and its paid up capital is Rs. 100,000.It is involved in Software publishing, consultancy and supply [Software publishing includes production, supply and documentation of ready-made (non-customized) software, operating systems software, business & other applications software, computer games software for all platforms. Consultancy includes providing the best solution in the form of custom software after analyzing the users’ needs and problems. Custom software also includes made-to-order software based on orders from specific users. Also, included are writing of software of any kind following directives of the users; software maintenance, web-page design].  
  
Aranya Consulting Private Limited's Annual General Meeting (AGM) was last held on 30 September 2016 and as per records from Ministry of Corporate Affairs (MCA), its balance sheet was last filed on 31 March 2016.  
  
Directors of Aranya Consulting Private Limited are Manish Kumar and Shikha Gupta.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| S No. | Topic | Page No. |
| 1 | Introduction | 4 |
| 2 | Technology Stack | 5 |
| 3 | Python | 6 |
| 4 | Django | 7 |
| 5 | Basic Environment Setup | 8-9 |
| 6 | Modules | 10-17 |
| 7 | Working & UI | 18-21 |
| 8 | Django Administration | 22-26 |
| 9 | Advantages of URL Shortener | 27-29 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**INTRODUCTION**

**URL shortening** is a technique on the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web) in which a [Uniform Resource Locator](https://en.wikipedia.org/wiki/Uniform_Resource_Locator) (URL) may be made substantially shorter and still direct to the required page. This is achieved by using a [redirect](https://en.wikipedia.org/wiki/URL_redirection) which links to the [web page](https://en.wikipedia.org/wiki/Web_page) that has a long URL.

For example, the URL "http://example.com/assets/category\_B/subcategory\_C/Foo/" can be shortened to "https://example.com/Foo", and the URL "http://example.com/about/index.html" can be shortened to "https://goo.gl/aO3Ssc". Often the redirect [domain name](https://en.wikipedia.org/wiki/Domain_name) is shorter than the original one.

A [friendly URL](https://en.wikipedia.org/wiki/Semantic_URL) may be desired for messaging technologies that limit the number of [characters](https://en.wikipedia.org/wiki/Character_(computing)) in a message (for example SMS), for reducing the amount of typing required if the reader is copying a URL from a print source, for making it easier for a person to remember, or for the intention of a permalink. In November 2009, the shortened links of the URL shortening service [Bitly](https://en.wikipedia.org/wiki/Bitly) were accessed 2.1 billion times.

Other uses of URL shortening are to "beautify" a link, track clicks, or disguise the underlying address. Although disguising of the underlying address may be desired for legitimate business or personal reasons, it is open to abuse.[]](https://en.wikipedia.org/wiki/URL_shortening#cite_note-twitter-2) Some URL shortening service providers have found themselves on spam blacklists, because of the use of their redirect services by sites trying to bypass those very same blacklists. Some websites prevent short, redirected URLs from being posted.

**TECHNOLOGY STACK**

* Programming language- Python
* Web Framework- Django-1.10
* Front End- HTML, BOOTSTRAP, CSS
* System requirement- Linux Operating System
* Text Editor- Sublime Text
* IDE- PyCharm
* Python

Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale.

Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library.

Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

Python is a [multi-paradigm programming language](https://en.wikipedia.org/wiki/Multi-paradigm_programming_language): [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming) and [structured programming](https://en.wikipedia.org/wiki/Structured_programming) are fully supported, and many language features support [functional programming](https://en.wikipedia.org/wiki/Functional_programming) and [aspect-oriented programming](https://en.wikipedia.org/wiki/Aspect-oriented_programming) (including by [metaprogramming](https://en.wikipedia.org/wiki/Metaprogramming) and [metaobjects](https://en.wikipedia.org/wiki/Metaobject" \o "Metaobject) (magic methods). Many other paradigms are supported via extensions, including [design by contract](https://en.wikipedia.org/wiki/Design_by_contract) and [logic programming](https://en.wikipedia.org/wiki/Logic_programming).

Python uses [dynamic typing](https://en.wikipedia.org/wiki/Dynamic_typing) and a mix of [reference counting](https://en.wikipedia.org/wiki/Reference_counting) and a cycle-detecting garbage collector for [memory management](https://en.wikipedia.org/wiki/Memory_management). An important feature of Python is dynamic [name resolution](https://en.wikipedia.org/wiki/Name_resolution_(programming_languages)) ([late binding](https://en.wikipedia.org/wiki/Late_binding)), which binds method and variable names during program execution.

* Django

**Django** is a free and open-source web framework, written in Python, which follows the model-view-template (MVT) architectural pattern. It is maintained by the Django Software Foundation (DSF), an independent organization established as a 501(c)(3) non-profit.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "pluggability" of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

Some well-known sites that use Django include the Public Broadcasting Service, Instagram, Mozilla, The Washington Times, Disqus, Bitbucket, and Nextdoor.It was used on Pinterest, but later the site moved to a framework built over Flask.

For developing a Django project, no special tools are necessary, since the source code can be edited with any conventional text editor. Nevertheless, editors specialized on computer programming can help increase the productivity of development, e.g. with features such as syntax highlighting. Since Django is written in Python, text editors which are aware of Python syntax are beneficial in this regard.

Integrated development environments (IDE) add further functionality, such as debugging, refactoring, unit testing, etc. As with plain editors, IDEs with support for Python can be beneficial. Some IDEs that are specialized on Python additionally have integrated support for Django projects, so that using such an IDE when developing a Django project can help further increase productivity.

**Basic Environment Setup**

* **Installing Python**

Django is written in 100% pure Python code, so you'll need to install Python on your system.

* **Installing Django**

Installing Django is very easy, but the steps required for its installation depends on your operating system. Since Python is a platform-independent language, Django has one package that works everywhere regardless of your operating system.

You can download the latest version of Django from the link [http://www.djangoproject.com/download](https://www.djangoproject.com/download/).

* **Database Setup**

Django supports several major database engines and you can set up any of them based on comfort.

* **Web Server**

Django comes with a lightweight web server for developing and testing applications. This server is pre-configured to work with Django, and more importantly, it restarts whenever you modify the code.

However, Django does support Apache and other popular web servers such as Lighttpd.

A project is a sum of many applications. Every application has an objective and can be reused into another project, like the contact form on a website can be an application, and can be reused for others.

* **Creating Application**

$ python manage.py startapp myapp

You just created myapp application and like project, Django create a “myapp” folder with the application structure −

myapp/

\_\_init\_\_.py

admin.py

models.py

tests.py

views.py

**\_\_init\_\_.py** − Just to make sure python handles this folder as a package.

**admin.py** − This file helps you make the app modifiable in the admin interface.

**models.py** − This is where all the application models are stored.

**tests.py** − This is where your unit tests are.

**views.py** − This is where your application views are.

## Get the Project to Know About Your Application

At this stage we have our "myapp" application, now we need to register it with our Django project "myproject". To do so, update INSTALLED\_APPS tuple in the settings.py file of your project (add your app name) −

INSTALLED\_APPS = (

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

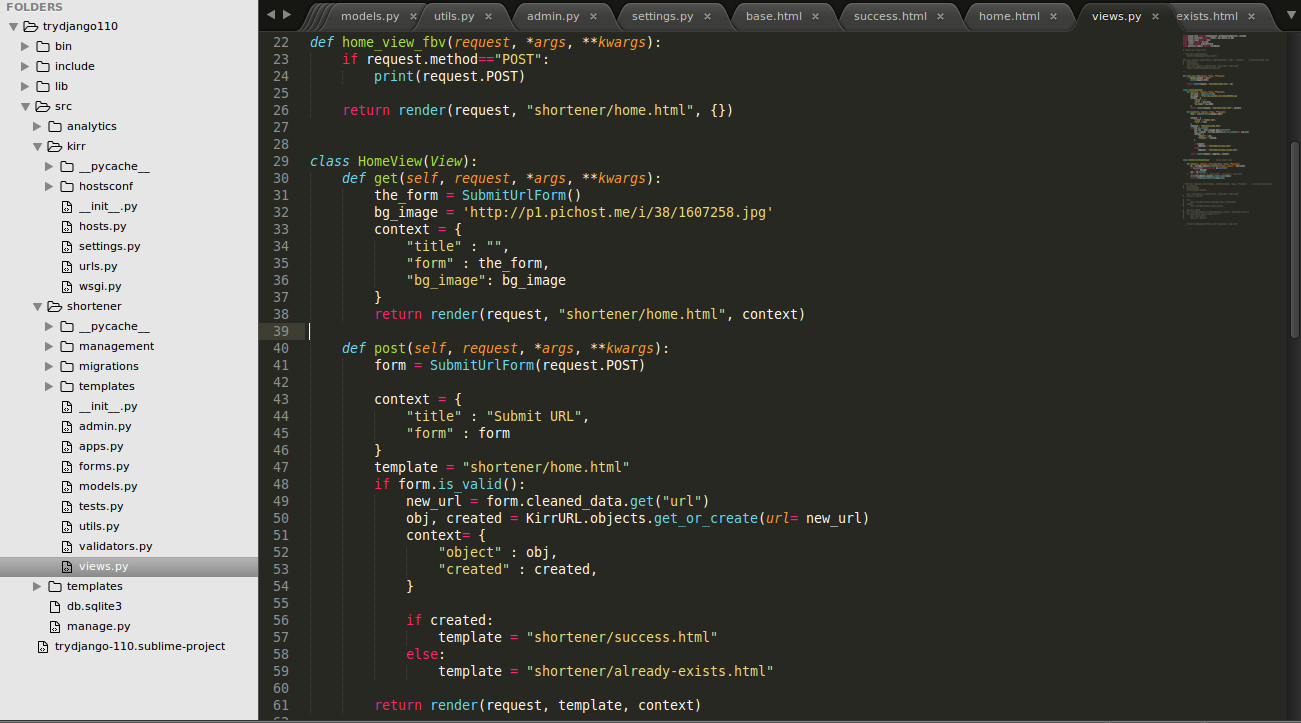
'django.contrib.staticfiles',

'myapp',

)

**Modules**

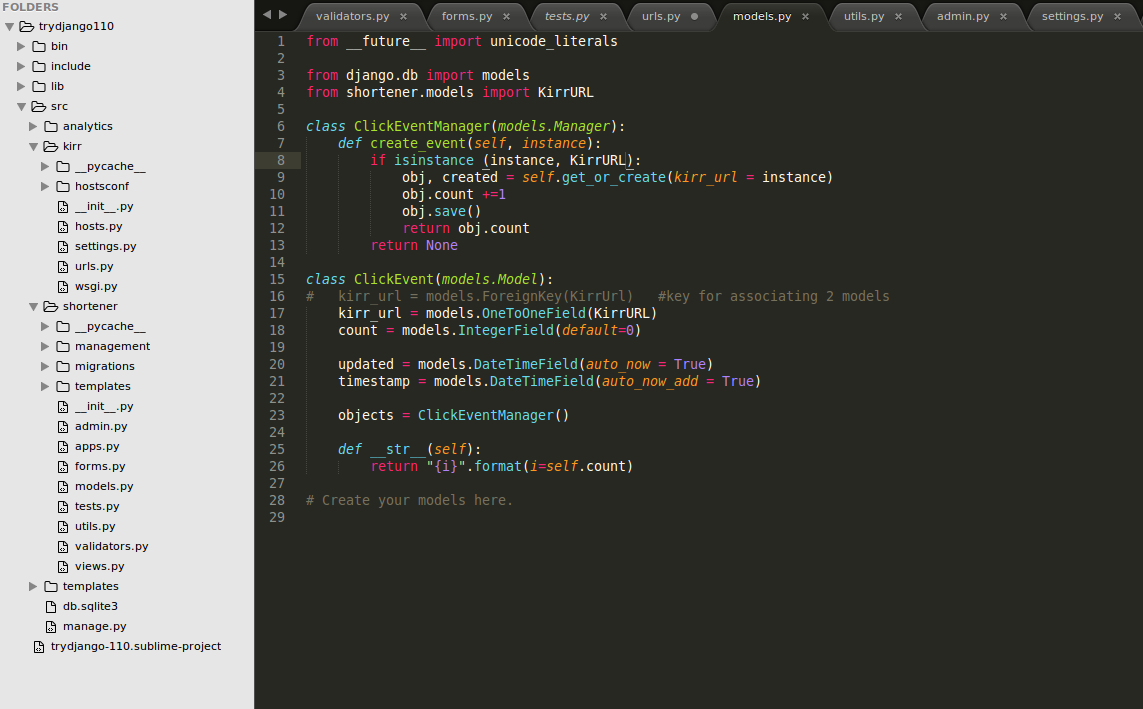
* Views.py

****

This module defines the function based views and class based views for the shortening service.  
Get and Post methods are defined here in this module. This redirects to either ‘success’ page or to ‘already exists’ page as per the type of input. If there is an input of URL which has already been shortened, it shows us the ‘already exists’ page whereas if the input URL is new, it shows us the ‘success’ page.

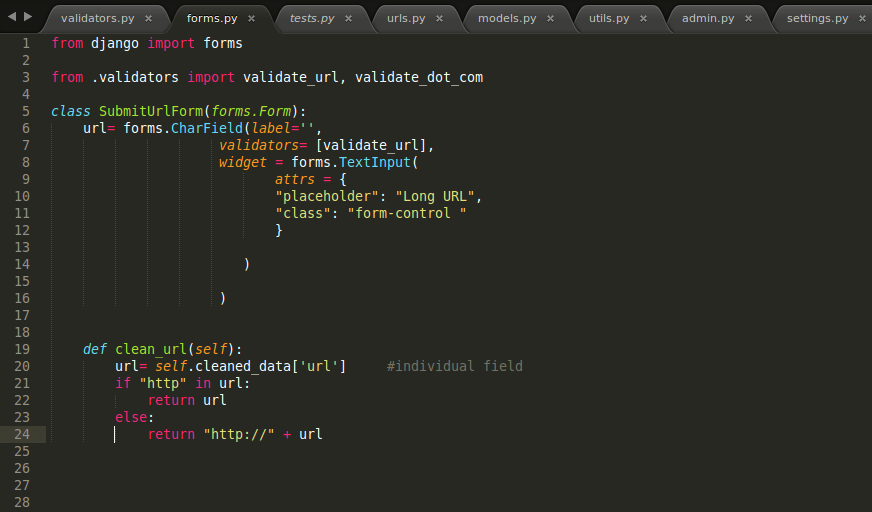
It also defines the home view; the background image, submit URL option are defined here.

* models.py



This python file defines the models for creating instances whenever a short URL is clicked. The object’s count value is incremented by according to number of clicks. The object is saved after every call of click event.

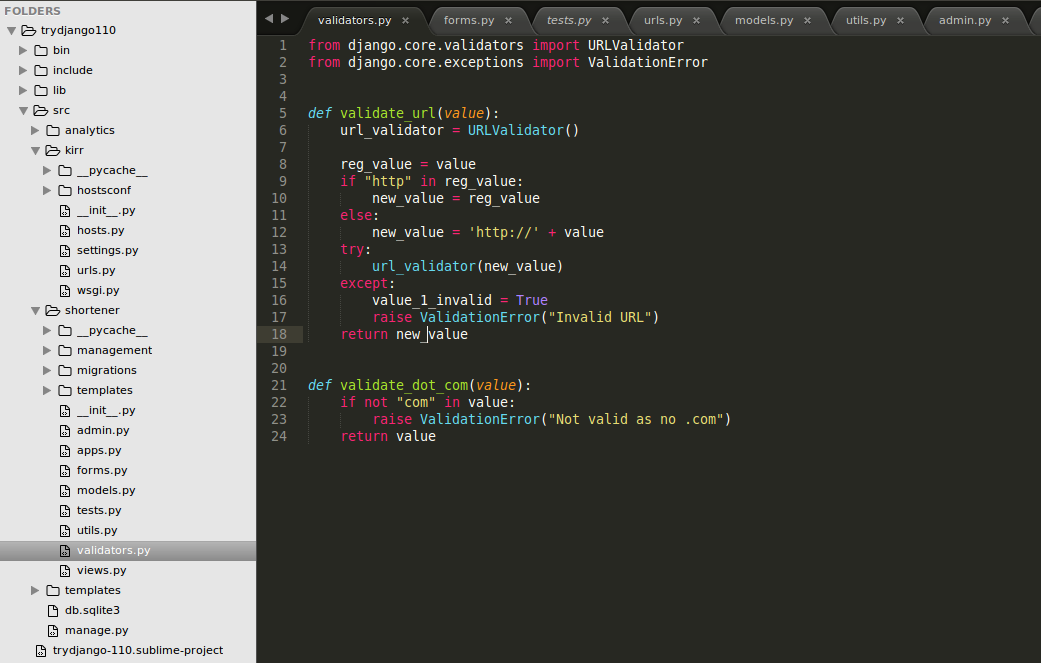
* forms.py



Forms.py deals with inputting long URL in the text field provided on the UI.

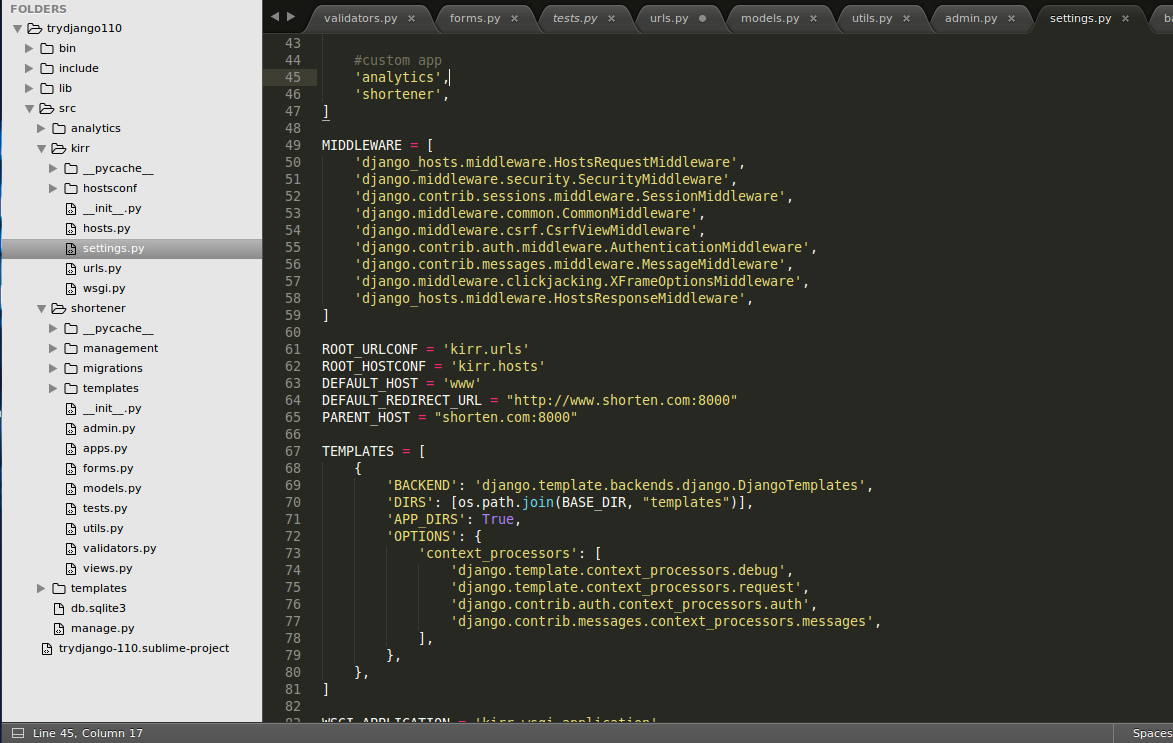
If the URL contains http:// it is directly stored otherwise it is added to the URL in the beginning to make it a valid URL and thus the data is cleaned.

* validators.py



Validators.py validates if the input URLs are valid or not.   
If a URL is not valid and is given as an input by the user in the placeholder provided a message is generated which shows that it is an invalid URL.   
The messages displayed are: “Invalid URL” or “Not valid as no .com” as per the failure in validation.

* settings.py



Settings.py deals with the basic settings in the shortener app; for example-

1. Default host
2. Default redirect URL
3. Parent host

It also defines which custom apps are added to the project like analytics and shortener.

* urls.py

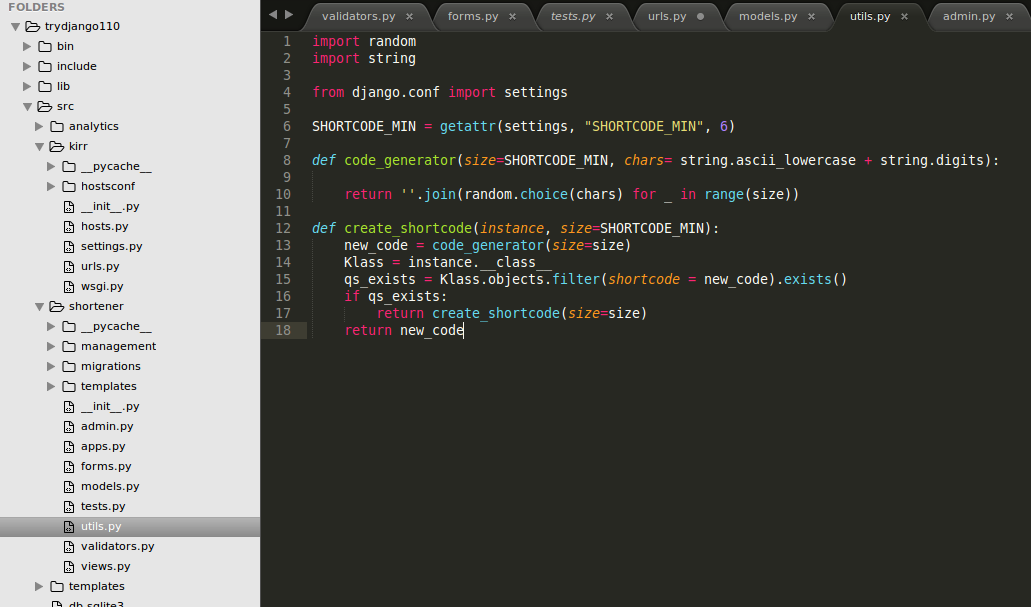


This defines the url patterns using regular expression for the urls.



This part in settings define the max and min length of short code to be generated. Here it is min=6 and max=15.

* utils.py



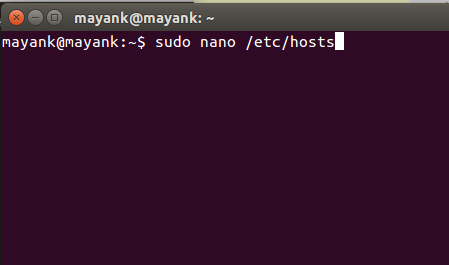
This is main function which generates the short URL for every URL whose short code has been requested.  
It is random combination of small case letters (a-z) and 10 digits (0-9)

This takes into account the maximum and minimum length of short URL generated and generates it accordingly. If the URL has already been shortened it returns the short URL previously assigned to it whereas if a new request is made it returns a new short code for it and saves it into the database.

**Locally testing a Domain Name**

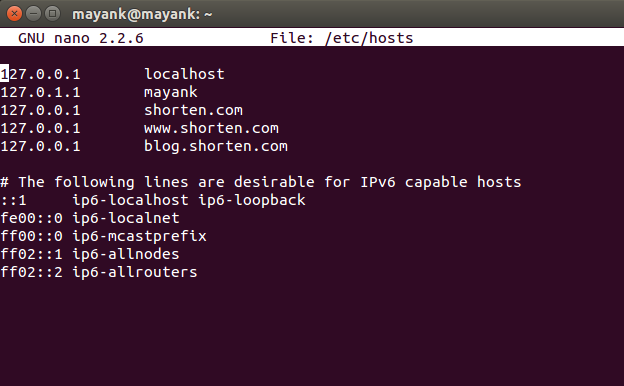
To locally test a domain, first we have to configure the Domain Name to the local server of our system.

Type in the following command in the terminal



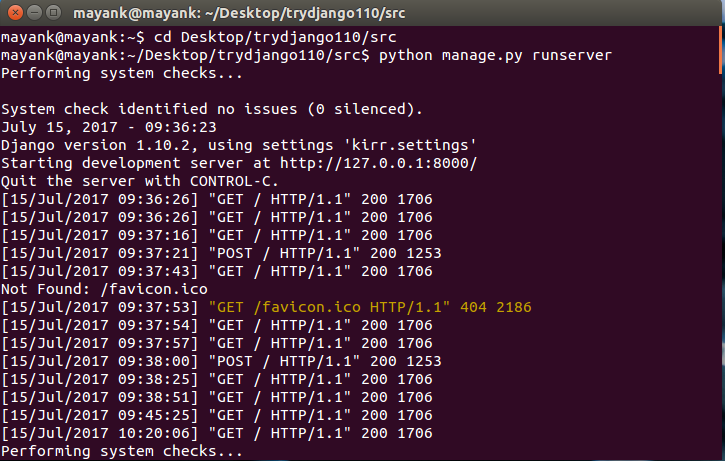
It will show all the configured domain names on the system and name one according to the need on 127.0.0.1 (shorten.com/blog.shorten.com etc.).

Multiple Domain Names can be configured to same address as they will all redirect to same link.



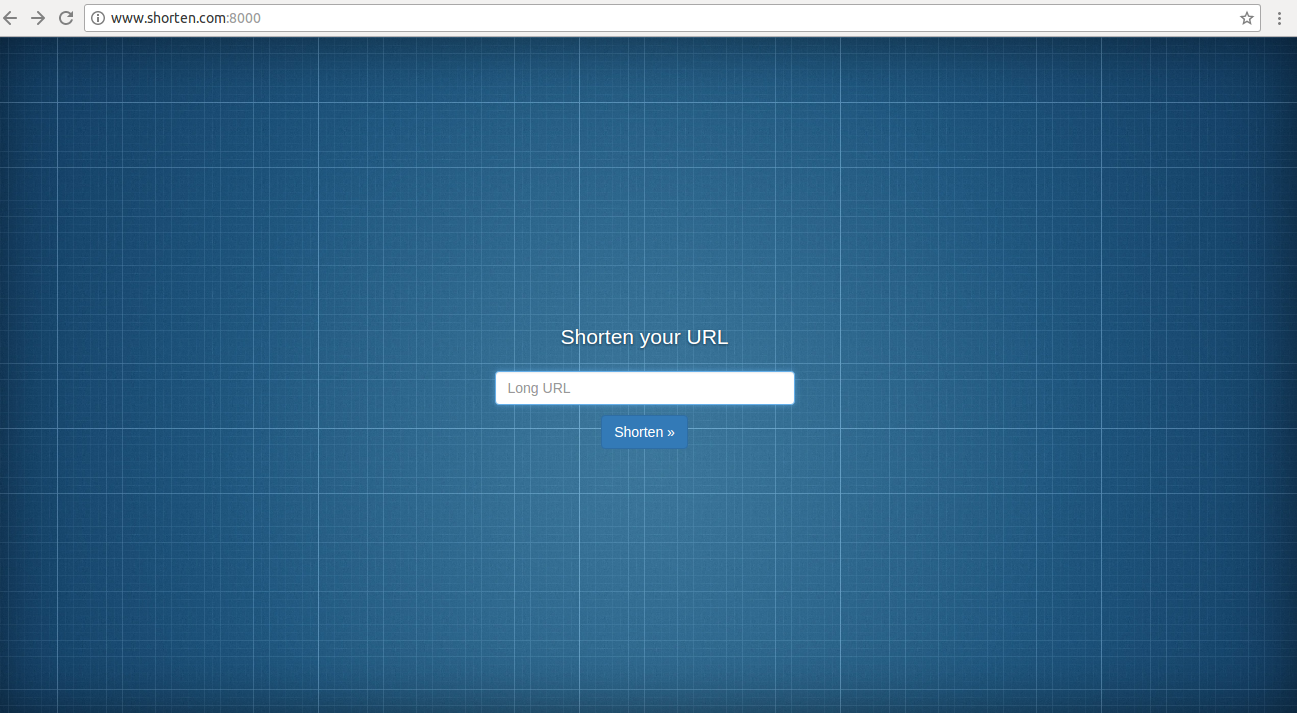
**Working & UI**

* To begin testing, start the local server using the following command.

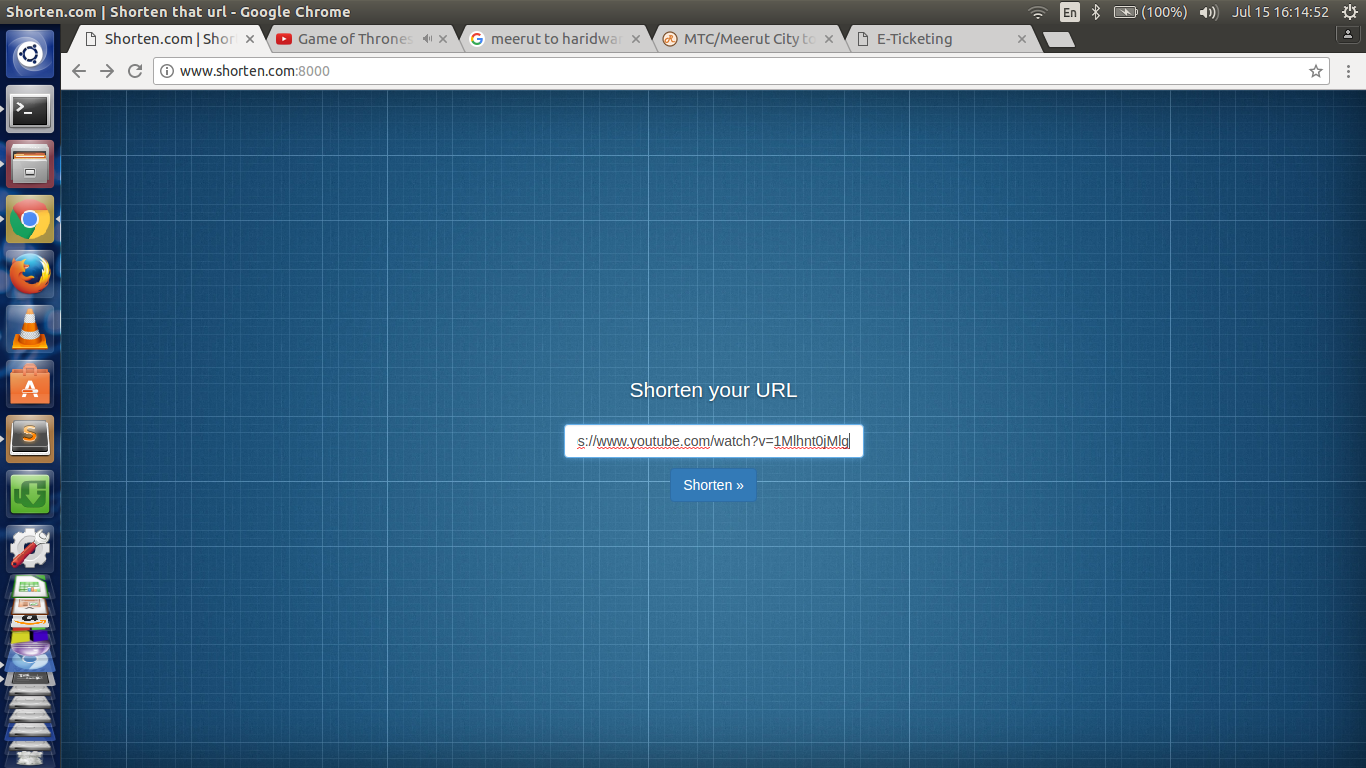
****

Server successfully started!

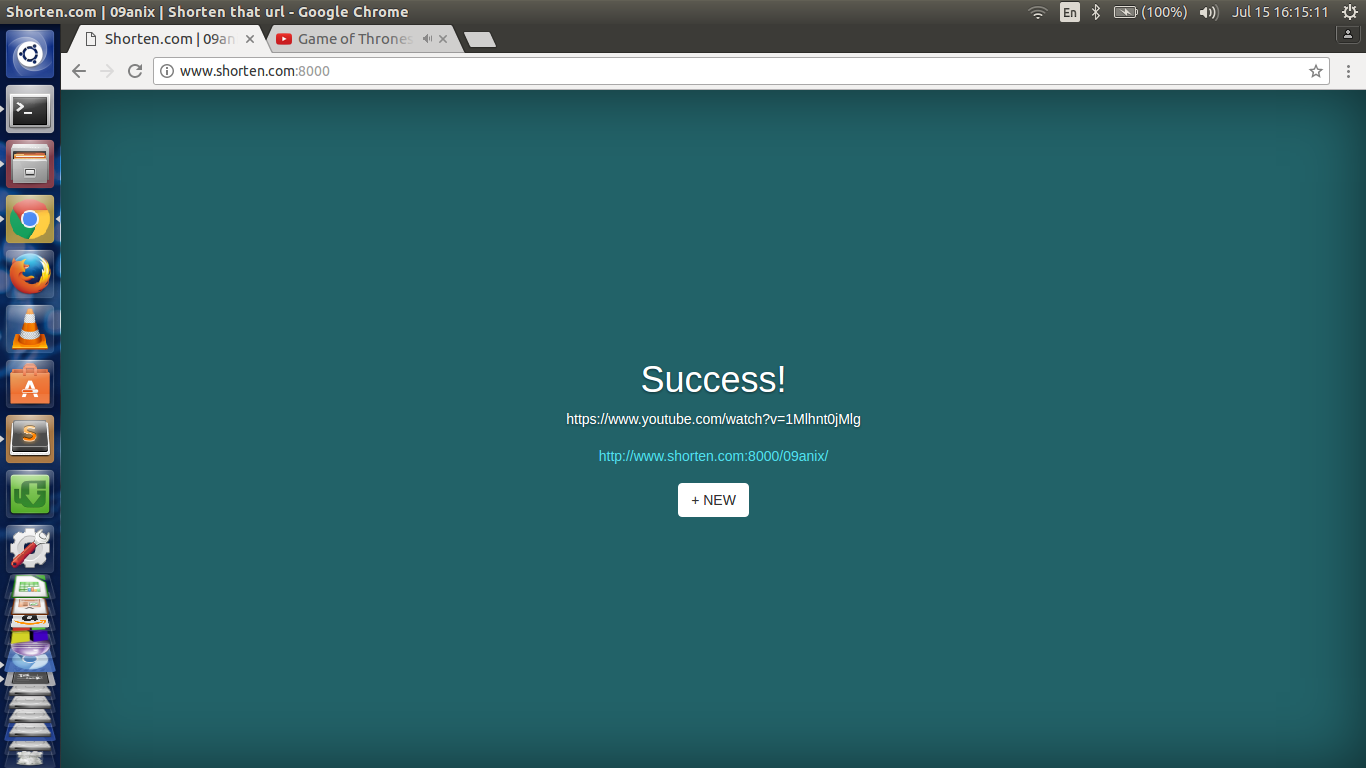
* Now type in the URL- shorten.com:8000 in your browser and the link will open
* This is how the start page will look like

****

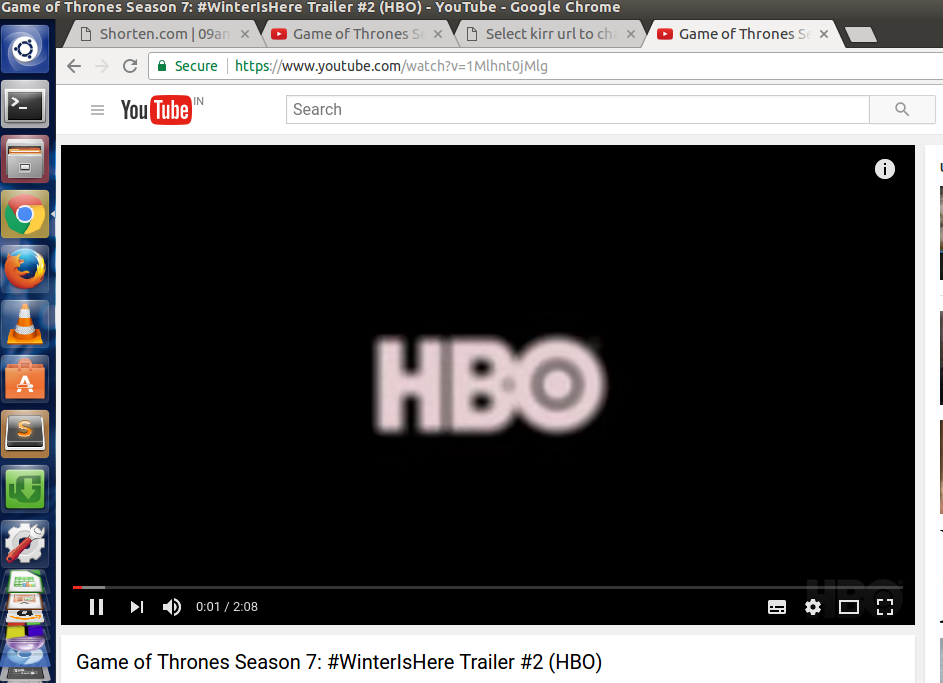
* Long URLs can be submitted in the space provided

****

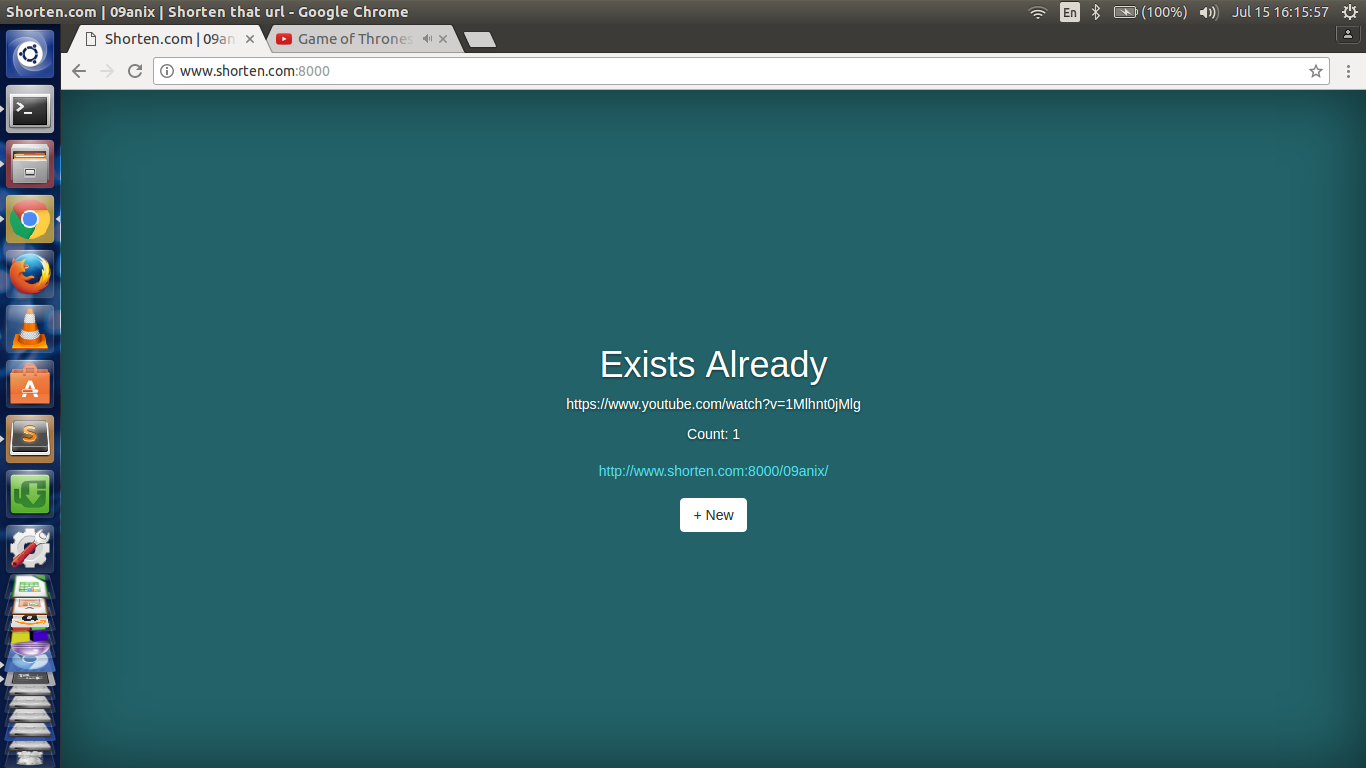
* If the entered URL is new to the database, it will generate the short URL for it and a success message will be shown.

****

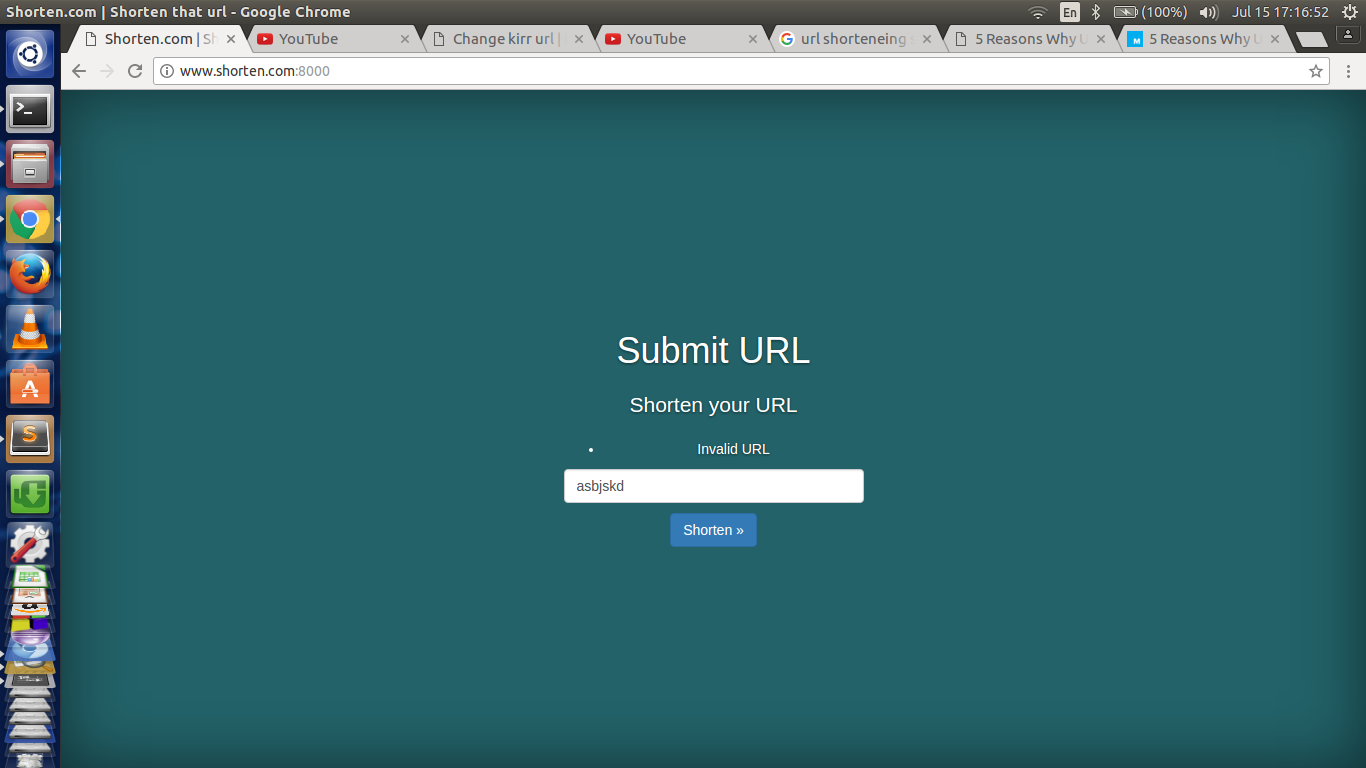
* Now the new, short URL when opened will redirect to the original URL which was entered by the user.

****

* If the same URL is again submitted to generate a short URL, it will display an ‘already exists’ message and will also show the number of times the link has been clicked.

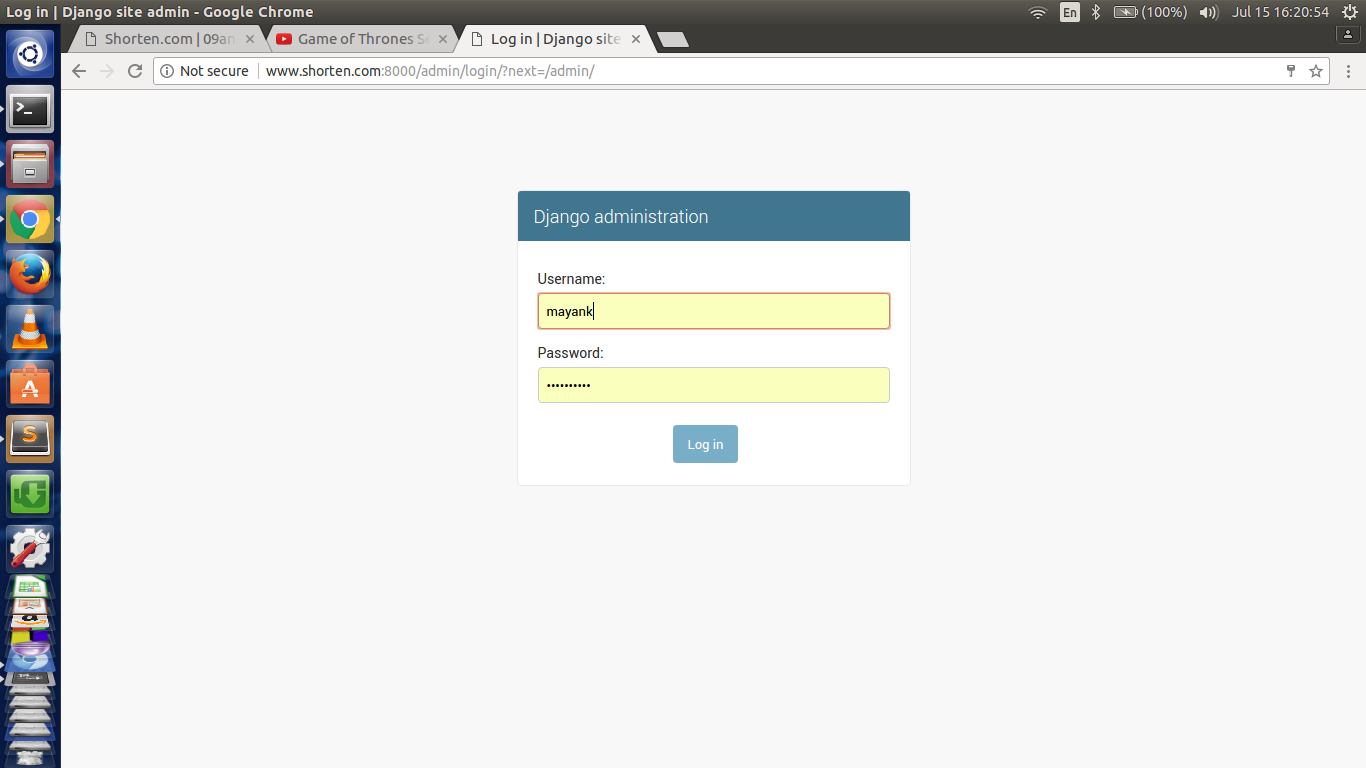
****

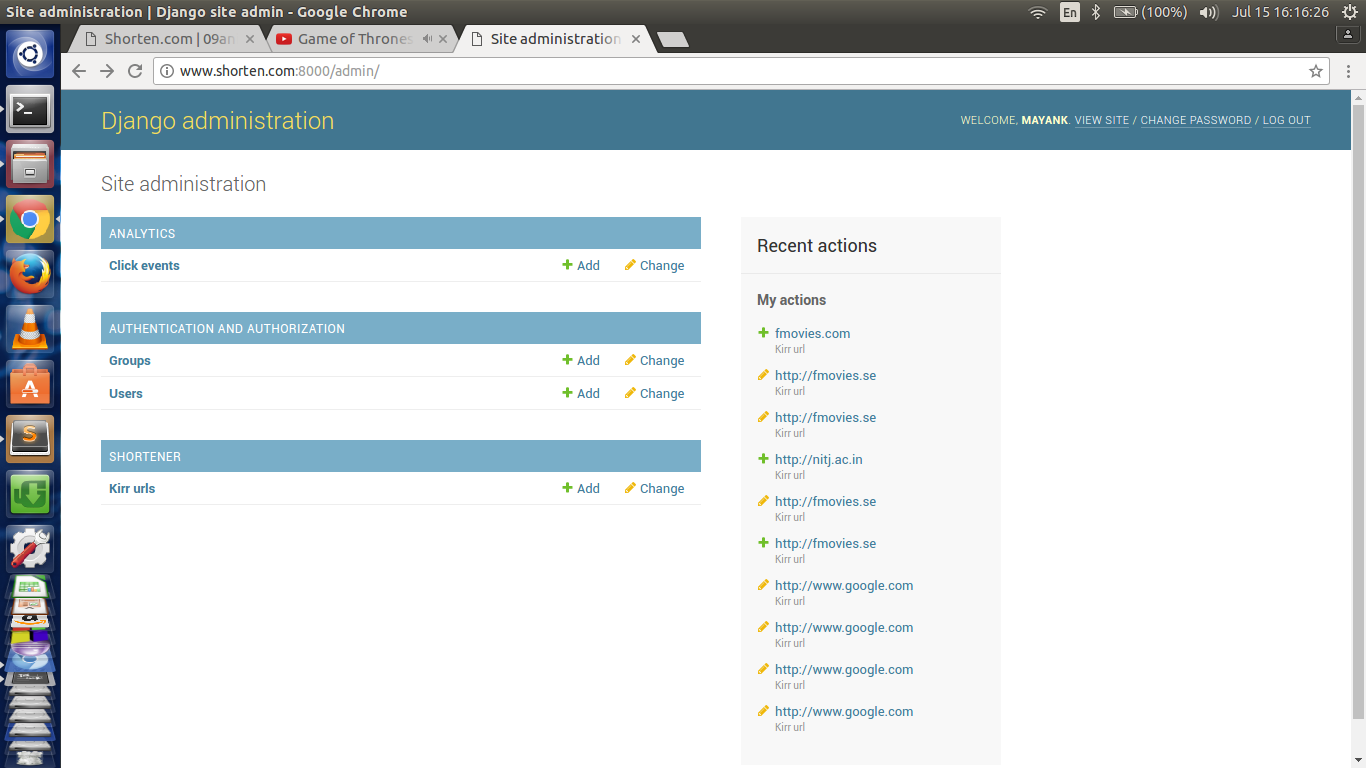
* If an invalid URL is entered a message will be shown indicating invalid URL.

****

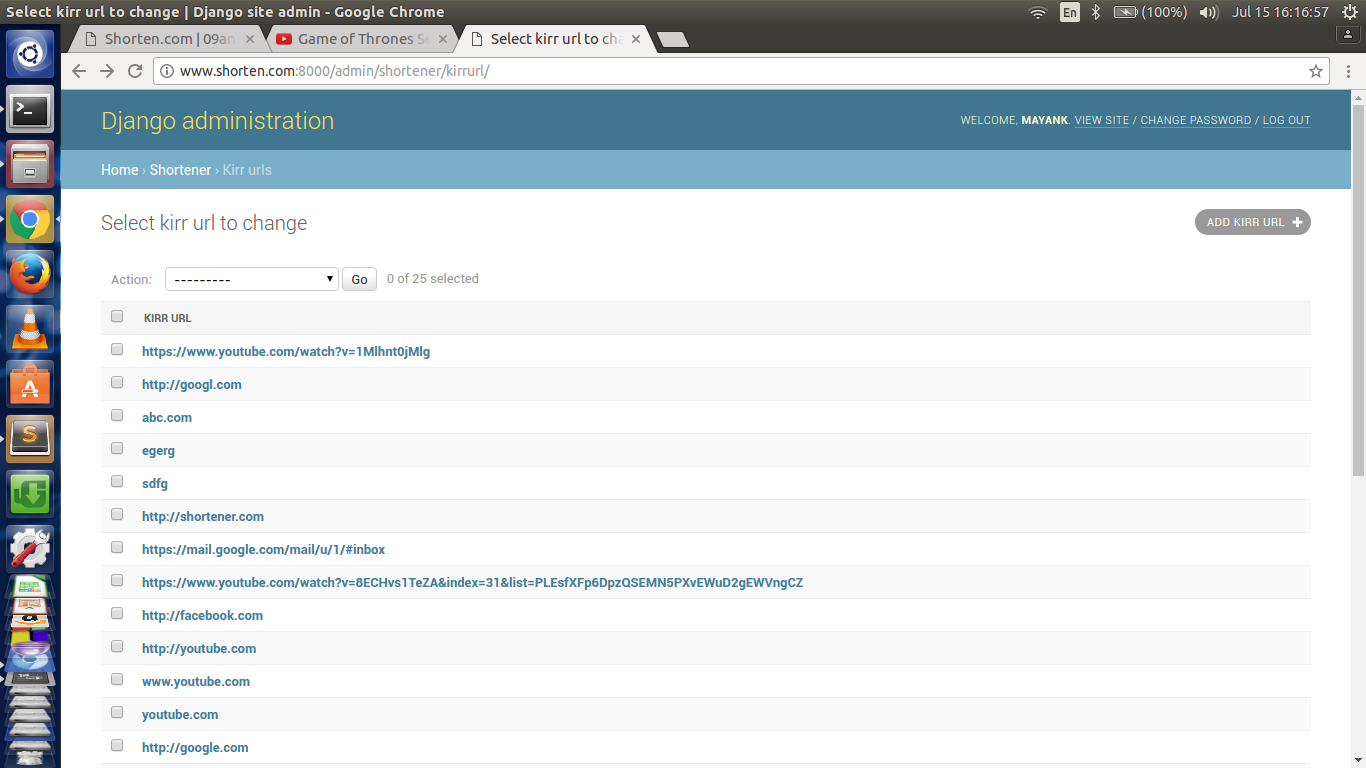
**Django Administration**

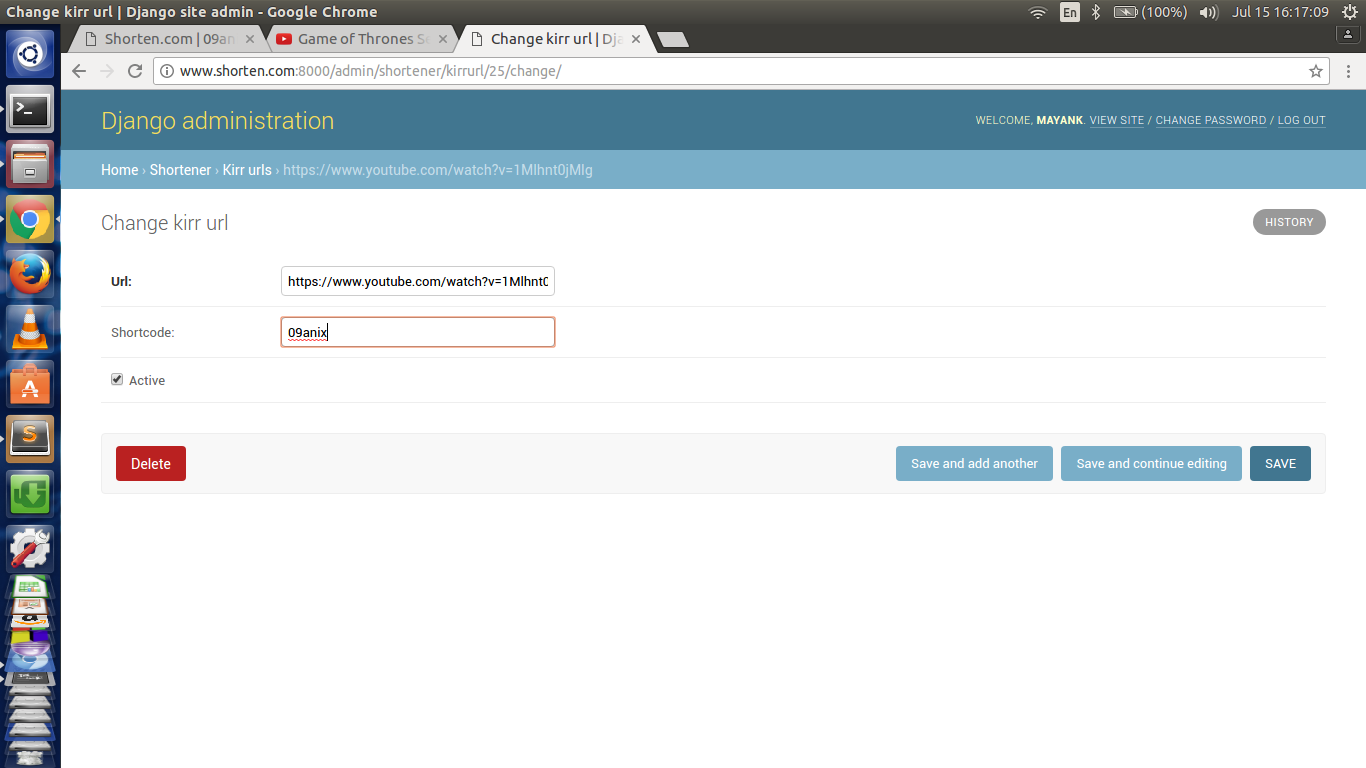
* A user can log in to Django-administration by entering the following URL- “shorten.com:8000/admin”. A login page will open user can be granted access by using the Login credentials provided in the beginning of Django setup.

****

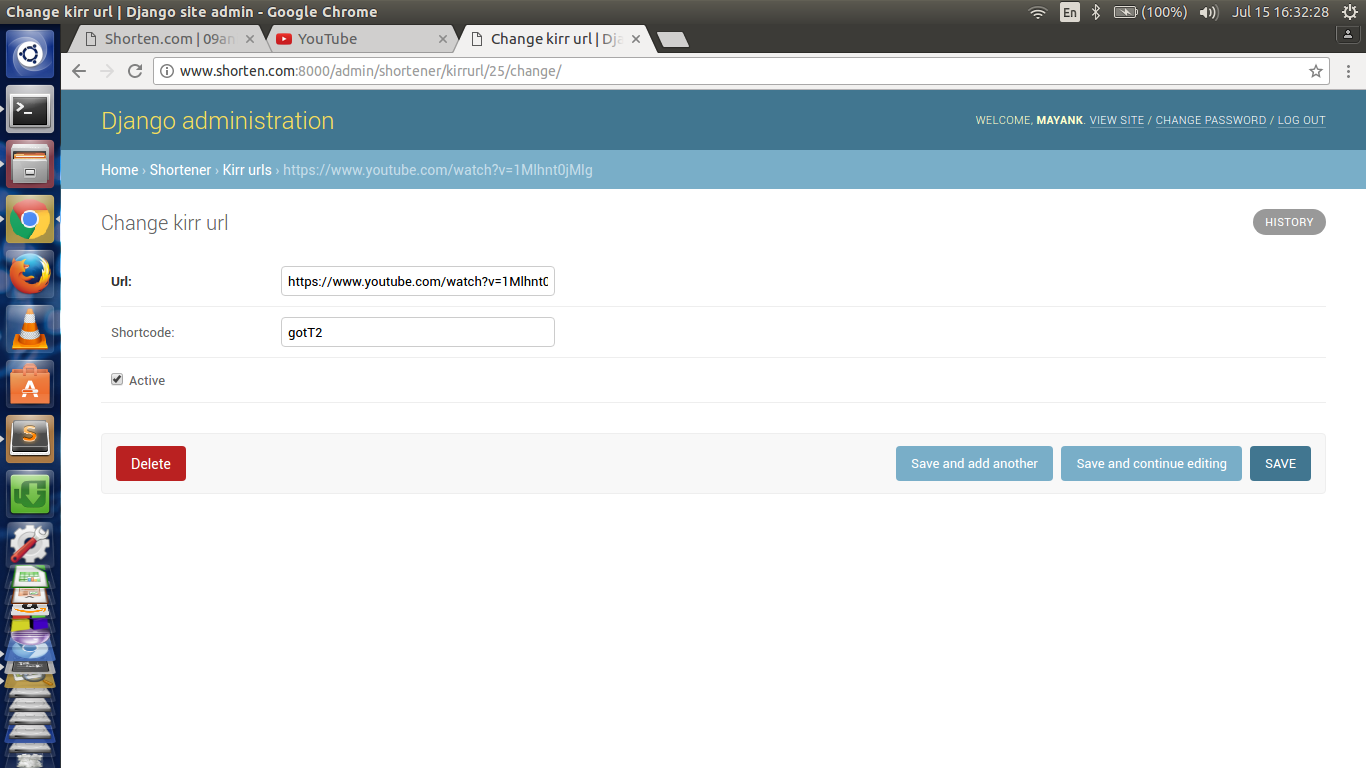
****

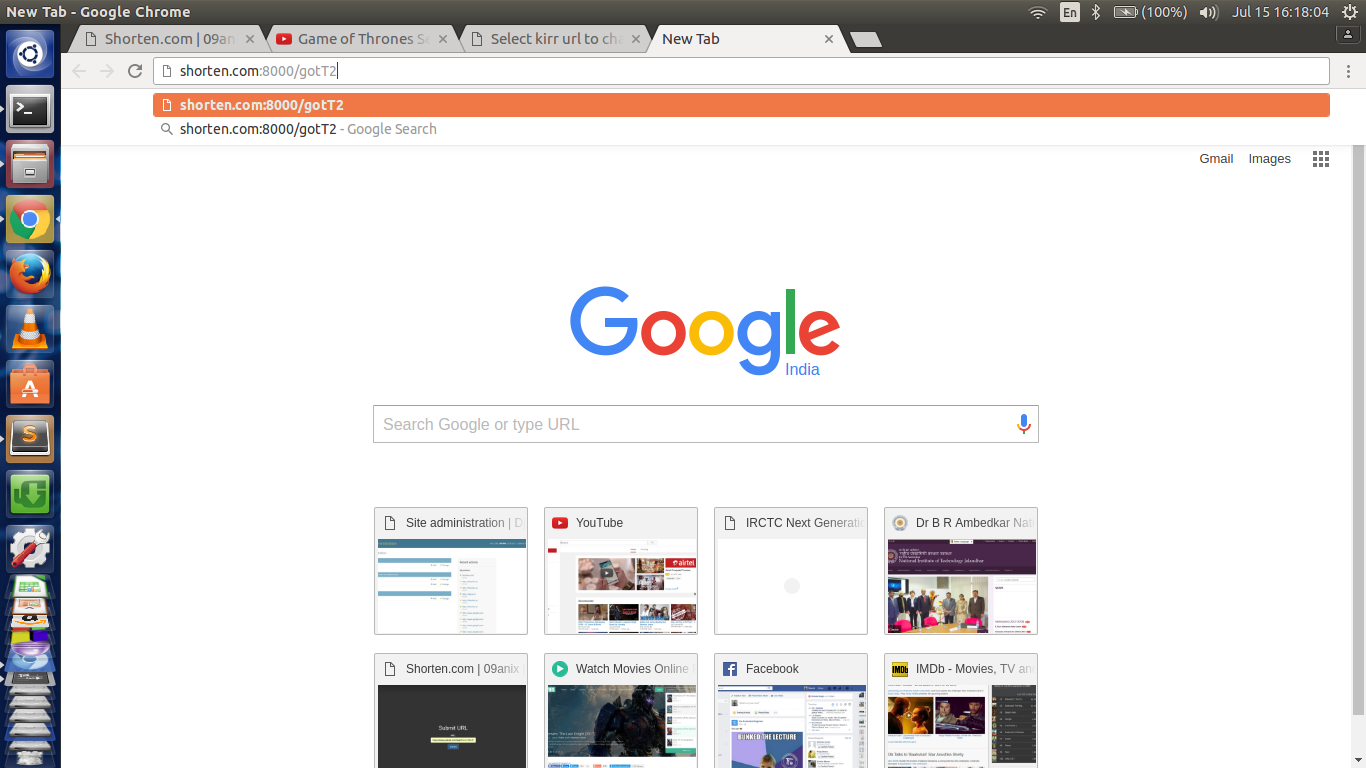
* Here user can manage his database. All the records of short and long URLs are here and the short URLs can also be modified as desired according to the convenience of the user. Records can be added, deleted, modified from here.

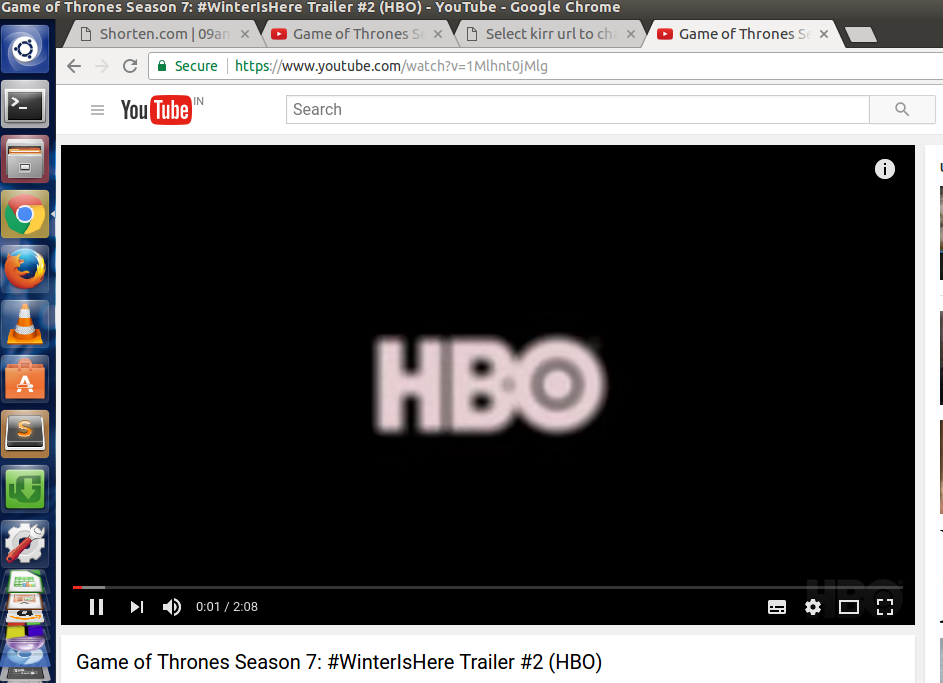




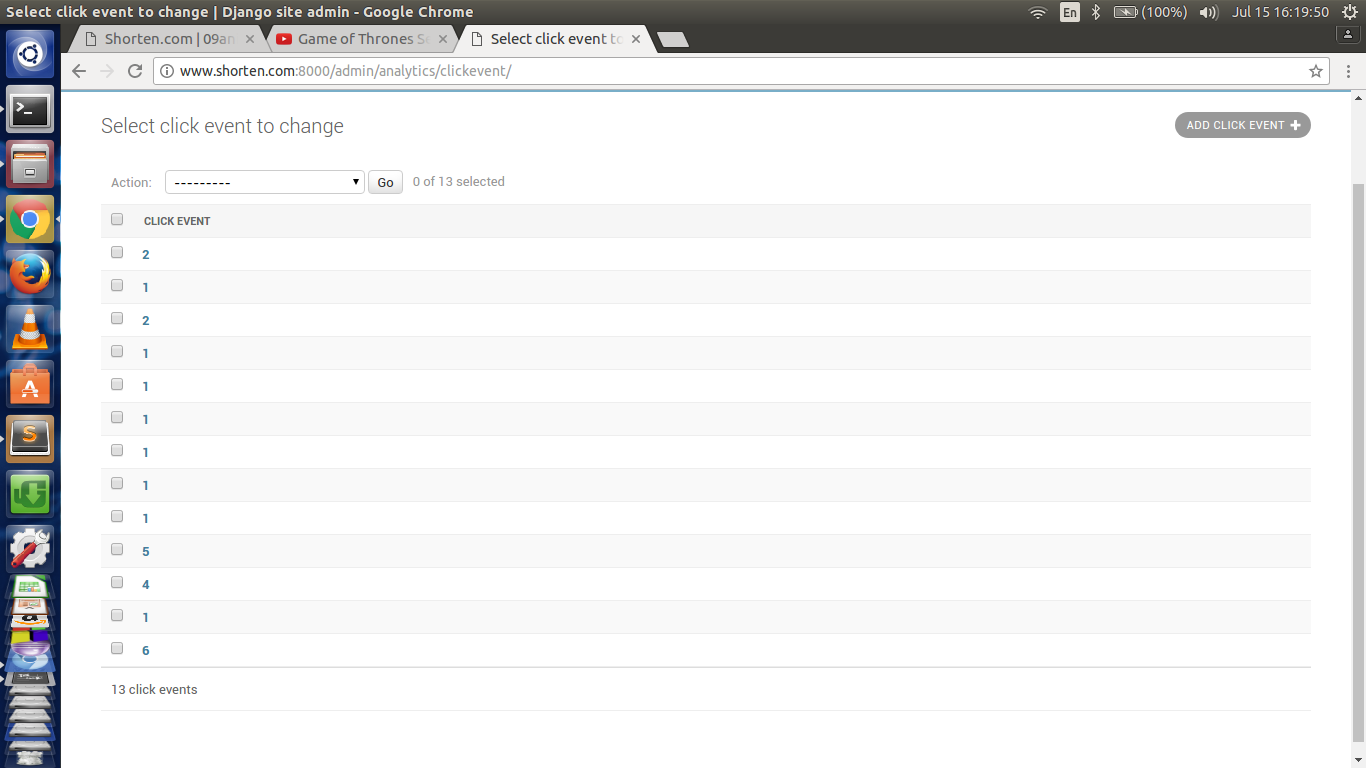
* The desired URLs can be edited and saved. Now the new short codes will be redirected back again to the original links. This can be used to easily memorize important links which are tough to remember and thus can be named according to the type of link.



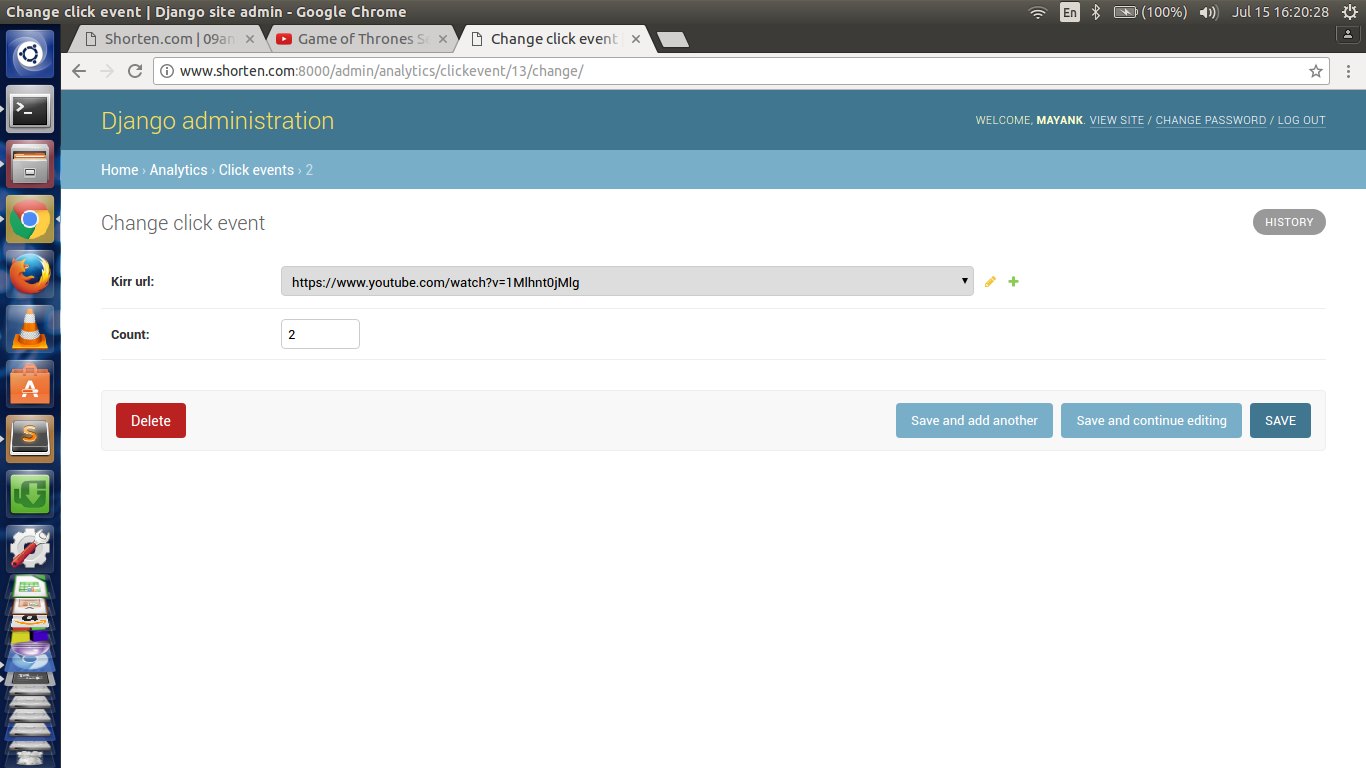




* Click Events can also be managed under the Analytics head.



* It will Show the number of clicks been made to every link present in the database.



**Advantages of URL Shortener**

**1. They make links more manageable**

What is easier to share?

**- This:** <http://friendfeed.com/e/b17ab777-ceb6-f7fd-8411-ca3a6ca9c9d4/Flutter-Mocks-You-Me-and-Our-Microblogging/>  
**- Or this:** <http://bit.ly/wPKks>

No, not all of the web's links are that messy. However, many links are long and wordy for SEO reasons. SEO, or search engine optimization, is about ranking higher in [Google](http://mashable.com/category/google/), and one of the factors Google and other search engines consider are keywords in the URL. This creates a conundrum for the user - the URLs help describe the content, but are lengthy and are not easy to share on emails, web pages, and especially social media services like [Facebook](http://mashable.com/category/facebook/) and Twitter.

URL shorteners help solve the problem of making links more manageable to share. And certain services, such as [➡.ws](http://tinyarro.ws/) allows users to add keywords to describe the link.

## 2. They can track and compile click data

One of the reasons that [bit.ly](http://bit.ly/) has received so much attention lately is because of the comprehensive data Bitly provides in the form of live click data, geographic location, the webpage the link where the link was clicked, and more. This type of information is invaluable to webmasters and companies - it shows where customers are coming from, when they are coming, and what interests them.

This type of information helps companies develop better products and webmasters produce more targeted content. Detailed information makes the economy more efficient.

## 3. They can be transformed into social media services

There was a good piece on GigaOM that discussed how Bit.ly could [launch its own version of Digg](http://gigaom.com/2009/03/31/why-bitly-could-upstage-digg/). While this may or may not be Bit.ly's eventual goal (just like Digg, a URL shortener could be gamed), it's clear that the data that URL shorteners can accumulate, coupled with the rise of short URL sharing on Twitter and other websites, could amount to some innovative social media services that display popular links, rank domains, and act as a filter or aggregate of social media content.

URL shorteners, in their own way, work as aggregates of information. This can lead to some useful mashups and innovations in how people share and digest content.

## 4. They can provide users useful features

A new URL shortener, [Pagetweet](http://pagetweet.com/) is a little more complicated than other URL shorteners, but when you look at an actual page via Pagetweet, it provides a useful interface for sharing via social media, seeing the number of views, and more.

Because it's so easy for companies to enter this space, innovations are constantly being made that improve the user experience. [Digg](http://mashable.com/category/digg/) recently launched a [DiggBar and Digg URL shortener](http://mashable.com/2009/04/02/diggbar-shorten-urls-and-experience-digg-on-any-web-page/), which provides information on the number of Diggs and comments any article has received. This is only the beginning.

Users also have a choice - they can follow links that will provide features, or simply choose ones with only the necessary functionality to get them to a web page. All in all though, URL shorteners can improve the browsing experience.

**5. They promote sharing**

You can simply fit more links and content in less space with URL shorteners. A tweet can describe and then link to a webpage in under 140 characters, while a full URL might not even come with an explanation.

Even more important is the rise of mobile smartphones, texting, and mobile Internet - it's far easier to text in a short URL than a long one. As Twitter, social media, and mobile Internet become more popular, the need to make sharing web content easier will increase. Shorter URLs are becoming more and more integral to that cause.

**6. Higher Click Through Rate**

People are used to seeing shortened links on social media. Due to this, using shortened links will increase your click through rate. It is one of the simplest changes you can make to your posting strategy in order to increase the effectiveness.

When it comes to social media, shortened URLs are the only kind of URLs.