URL SHORTENER PROJECT REPORT

To do: Create a URL shortener app.

Tech Used:

Front end: HTML, CSS, Bootstrap

Backend: Flask

Database: SQLAlchemy

Approach:

* At first, I simply created the basic flask app for working, and later connected the main function with the index.html
* Then I created base.html, I used getbootstrap for some CSS, and also created index.html and extended base html in it.
* Now, I created the input form in index.html and history button in it.
* Created database url class, with the following:

Sno = serial no.

Web: input url

Short: shortened url

Date\_created: to show the date

* Created the constructor for this class
* Now in the home function, I took the requested url and generated special code of length =16 using the random.choices() function.
* Checked if the taken input url is already present in the database, if it is redirect to shorten url, if not we will move on
* If the url in function is not already then we’ll generate the short url function, check if the short url is already made, if already made, we’ll make a shorten url function again.
* Now we’ll create a new object of Url class, and add the object to database and commit.
* When given input we’ll redirect to shorten.html which will show the new url.
* Now we’ll create a route for history, and write code to redirect to the history.html page with allurl.
* I’ll create the search function, so that we can search using the given short code. Here, first find the object using shorten and then redirect to the web using that object.
* Now, I will create the redirect\_new\_url function which will bring us to shorten.html page, here we can see our shorten html function.
* I also created the delete function, we’ll get the serial number from history page button where we will simply find the object. We’ll delete that object and redirect to history too.
* Write the app.run() now.

Now I will talk about the various html pages

* Here, I have four html pages.
* Shorten: this is just used to show the shorten code.
* Base: this page is just the base of other two and have navbar.
* Index: this page extends the base function. It takes url and have button for submission, this page also contains the history button.
* History: history page shows all the input url with their shortened ones. We can also see the date created in too.
* This page also contains two important functions/buttons , that is, delete and search:
* Delete: this button calls the delete function and therefore deletes the object using the serial number.
* Search: this button call the search function and redirect us to the long url.

How I generated the shortened URL:

* At first, I created a list which will contain the shortened URLs to check if some url is repeated not.
* Then I used a while loop with true in it, if I find the proper random url, then I’ll break out of the url
* Then I used choices function fron the random module, and provided it with ascii\_lowercase, ascii\_uppercase and digits to increase the randomness of the shortened urls.
* Now I checked if the url is present in shorts list, if it is, loop will carry on, and if not, we’ll break out of the loop.
* And with this way we get out our shortened code.

Code: while(True):

               short = ''.join(random.choices(string.ascii\_uppercase+string.ascii\_lowercase+string.digits,k=16))

               if short not in shorts:

                    break

            shorts.append(short)

Delete function: to delete the history related to any object, I used the delete function.

* The delete function call was done from history page.
* We find the object using query.filter\_by().first() function.
* Then, used the delete function from SQLAlchemy and used commit.
* Code:
* @app.route('/delete/<int:sno>')
* def delete(sno):
* url = Url.query.filter\_by(sno=sno).first()
* db.session.delete(url)
* db.session.commit()
* return redirect("/history")