WEB-APPLICATION: FINAL REPORT

INTRODUCTION

Christmas Wishlist is a web-application running with Python within the Levinux environment. The concept of this web-app is to ask users what would they like for Christmas and they could send their message to Santa. The web-app transformed the old Christmas letter on paper into a digital Christmas wish list that will make the user's life easier.

Users will need to login to start writing their wish-list. They are redirected on the main page of the app where they are able to submit their wish that will be directly added to the list display below. Another page is accessible for the users where they can listen to Christmas songs and send a personalize letter to Santa. Those different functionalities contribute to a festive atmosphere and make a better enjoyable visit. At the end of their visit, they can logout from their session and come back later to continue their wish-list.

DESIGN OF MY WEB-APP

The architecture of the application is composed with 7 different routes running through Python Flask in Levinux.

The first route defines the sign-up page where the user can create his profile that will be added to the database. The page displays a form that is connected to the database that will insert the user's details for each new request. Creating your account will be useful so the user can start his session on the web-app and come back later without losing all his information already submitted.

Users can then redirect to the second route, the login page, where they can connect to their profile already created in the page before. This page displays another form that will check if the user is already created. If the user is not stored in the database, he won't be redirected the main index page but to the same login page with an error message.

The third route describe the main index page of the application where users are able to add their wishes and see their list below. This page redirects three routes in a loop. The index route selects the items from the 'wishlists' table in the database to display them in the list. Those items are added by the users with the 'add' route. The 'remove' route will help the user manage his wish-list if an item wants to be removed from the database. Those three routes redirect the index page in a loop.

The sixth route redirect the user into a nice entertaining page where they are to listen to some Christmas music and send a personalize letter to Santa.

The final route requests the user to logout from their session and redirect them to the root of the application (the sign-up page).

For designing my web-app, I have used Bootstrap with Jinja2. I found a nice fluid bootstrap theme on startbootstrap.com. It makes my web-app more attractive for the user that he will like to stay longer to find out more information.

CRITICAL EVALUATION

To build my web-application, I looked for an original and useful subject that would not be too similar from all the other applications we can find on the web. I finished by creating a Christmas wishlist for innovating the concept of the old paper Christmas letter into a digital Christmas wish-list. It will more manageable and useful for users as they can change their ideas and keep their list clear.

When the user arrives on my main application, he is directed to the sign up form by giving a name, email and password before entering on the website. When he submits the form, he is redirected to the home page with a personal message with his name. The advantage of customize with the user's name make it more personal for the user as he receives a nice welcome message and attracts him to stay. This also means the user is creating a session where he can log out from it and log in again later. The user's details are stored in the database. The user is then able to connect automatically with his username and password on the logging page. The problem with the logging form is that even the users already stored in the database cannot access to the index page. Otherwise, users create another profile and will see the wish added from all the users. Some other research should be possible to work on to personalize your own list for each user so users can personalize their own list.

On the main index page, the user is able to add an item to the wish-list where the form redirects the same page with the item added in the list below. The user is also able to remove an item from the list. Those functionalities make the app running well and go straightforward to what the user needs.

To stay in the Christmas atmosphere, another page is created for keeping the user entertained with some music and personalize your own message for Santa Claus. The music has some short error with the JavaScript code. The link from the song do not recognize the JavaScript function when it is requested to be played, but otherwise the elements are well rendered and displayed as requested.

Moreover, I decided to use some red, green and white colours for my theme to influence a friendly atmosphere as users will stay and come back quite often to add their ideas to their list. So, it is important to make an attractive application. A background image is used to stay in the Christmas theme and all the content are writing in darker font on white background. The white

makes the application more relaxing to visit with lots of spaces between each paragraph. Darker fonts are used for better accessibility for disabled people on the application.

The only defaults of this application is that it would need to be more personal for each users and allow users to log into their account again without in the need of creating a new profile. It will then clear the database and make it more understandable for users. For the moment, users will only able to always create a new account but they can still see the item added to the database.

PERSONAL EVALUATION

What did I learn, challenges I faced, and methods I used to overcome them?

For the second web application we had to create with Python-Flask, I have seen lots of challenges from the first coursework I haven't been able to manage, but I did manage to find a method in the second assignment.

As for example, I practised creating several databases in Python using sqlite3. With this database, I was able to create two different tables. One for the first form on the logging page and the second one for the wishlist. On the logging page, the user is able to create his own profile and his details will be stored in the database, but I still have an issue when the user wants to logging again using the login page. For being to access the page, I have put in comment my work that does not work for the moment and I have only added one access with this form. Otherwise, users can create another profile and they will be access to the index page.

The second table "wishlist" from the database makes more effect on the application as I managed to make all the routes working in Python. Users will be able to add an item to the list and it will be inserted as the same time in the database. With this item, I manage to select it and display all the elements in a list on the index page and another functionality I managed was to remove an item from the list. All these functionalities were a challenge. For the remove function, I have forgotten to request the actual URLs parameter so at the beginning it did not remove the item from the database but the form was still answering as it was deleted.

This index page has been a lot of challenge to make everything work properly that I was more on focus on those problems to fix. As the second page I created still need a bit of work as sometime when I click on the music page, it breaks the code but I preferred to focus on the main functionality that user is looking on the application. On the second page, the user is still able to send a fake email to Santa.

In conclusion, I can prove I have learned and did a lot of progress since the first coursework. Although, there is still need some concentration on a few point to make a good professional application. Creating a web application from scratch is difficult and Python is a sensible language that needs a lot of attention on your code as it can break very easily.

SUMMARY

REFERENCES

https://startbootstrap.com/template-overviews/business-casual/

https://www.youtube.com/watch?v=rHGPQj8sfUs

https://realpython.com/blog/python/introduction-to-flask-part-2-creating-a-login-page/

https://github.com/realpython/discover-flask

http://flask.pocoo.org/docs/0.11/tutorial/views/

http://sqlitebrowser.org

https://www.tutorialspoint.com/flask/flask_sqlite.htm

http://codereview.stackexchange.com/questions/110679/simple-login-system-using-python-flask-and-mysql

https://bootniyas.wordpress.com/2014/07/23/student-record-app-using-flask/