CEASER CHIPER WEB APPLICATION

A single html web application where the user enters an input string and encrypts it by shifting its characters by the English alphabet. The shift amount is determined by the user.

Front-end is developed using angularjs, html and bootstrap. Back-end is developed in python. Communication is done by json.

HOW TO RUN APP:

1- Unzip files
2- Run app.py.
3- From browser, navigate to url: *localhost:5050*

USER VIEW:

INPUT TEXT FIELD: Text area where the user enters the text he/she wishes to encrypt.

ROT: Number input field where the user chooses the shift amount for encryption. Only numbers are allowed in this field.

BUTTONS: Used to encrypt and unencrypt the texts. Right button is to encrypt from input to output while left is to decrypt the text in the output field.

OUTPUT TEXT FIELD: Result of the encryption is shown here. The field is read-only

CHARACTER CHART: The chart where the number of characters in the input area and in the output area are displayed.

WARNING MESSAGE: If the user fails to encrypt the input due to wrong rotation(such as 0 or 26) a warning message is displayed.

TECHNICAL:

FRONT-END

In front-end Angular 1.5.8 is used with html and bootstrap. In templates folder html file is stored. In Static folder javascripts can be found.

Templates/

CeaserChiper.html: The content with chart, text and input & output fields

Static/

Angular-chart.min.js & Chart.min.js: Required javascript files for chart creation

CeaserChiperApp.js: The javascript file for establishing communication with back-end, creating the chart and producing the warning message.

- Uses the http GET method to make the html rendered
- Uses http POST method to send json containing text, rotation number and operation type to back-end. As a result receives a json response which contains the result string.
- Produces and manipulates data for the chart.
- Checks if the encryption is successful. If not, produces a warning message

BACK-END

App.py: Flask framework is used to manage the web application routing. Json library is used to parse and produce json.

- Webapp is deployed on the server: *localhost:5050*
- GET and POST requests are handled in the home function. By GET, the html page is rendered. By POST, json input is parsed and after operations(encryption or encryption) json output is returned as a response
- Encryption and unencryption operations are handled in encrypt and unencrypt functions.