1. How can we enhance the backend to have faster performance for frequently accessed “short URLs”?

    a. Cache as much as possible.

    b. Reduce the number of HTTP requests.

    c. Load JavaScript asynchronously.

    d. Improve SQL Query Performance.

    e. use another servers

2. How can we enhance the implementation to have one “short URL” redirect to a few “long URLs”, based on redirect percentage? For example: for http://localhost/sh54

a. 30% of the redirections will go to www.somethingverylong.com/1234.html

b. 70% of the redirections will go to www.somethingverylong.com/3456.html

its cant happen because the “short URL “ just have one “long URL “ .

3. How can we enhance the implementation to report the top 5 redirected URLs in the last hour?

the top 5 redirected URLs it mean the top redirect URL who get a most visited and i added something to know how many visit in redirect URL ...

i have a many solves for this problem but I think the better thing is to create a special database for all URLS in the last hour and this database refreshes all time and this database will be in binary file to saved places and to get data faster and to be easy to update it .

4. How can we prevent an attack, in which a malicious client fills the database with millions of URL redirections?

a. Validate input strings on the server side : Validation is the process of making sure the right type of input is provided by users and to neutralize any potential malicious commands that might be embedded in input string.
b. Use command parameters : A better alternative to escaping would be to use command parameters. Command parameters are defined by adding placeholder names in SQL commands, which will later be replaced by user input.
c. check the code of every page for places where you combine page contents, commands, strings.

5. What happens after the user presses the “Create short URL” button? Please detail as much as possible.

after the user presses the “Create short URL” button in backend it will go to add_link() and get original_url using request.form['original_url'] ,original_url is a input from page index , after that the URL send to models.py (formate for DB) and using a Link function , Link func open a new element for this orginal URL and define a many contains like : id , original_url, short_url, visits and date_created

id : integer num and primary key .

original_url : a URL who get from input

short_url : a String with four char .

visits : a integer num (counter how many visits in page) .

date_created : DateTime and get datatime now.

and add it in DB file .

and commit that .

and after that we go to link_added.html and put the orginal URL in his place and put short URL in another place.