Werkzeug

General Information & Licensing

Code Repository	https://github.com/pallets/werkzeug/
License Type	BSD-3-Clause License
License Description	• Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met.
License Restrictions	• Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. • Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. • Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
Who worked with this?	Collin and Jieli

generate_password_hash

Purpose

Replace this text with some that answers the following questions for the above tech:

- This tech provides us with the ability to easily salt and hash passwords that are given to us.
- This tech is specifically used in the register() function on line 91 in app.py. The generate_password_hash function call can be found on line 115 in app.py.

Magic

- This tech does what it does by taking in a plain text string, generating a salt and then computing the sha256 hash of the plain text input and the generated salt and returning a string in the form of f"{actual_method}${salt}${hashedPassword}. The reason why it is returned like this is so that developers are able to get all the proper information they need for password storing in one single function.

- The code can be found in the file security.py located here: https://github.com/pallets/werkzeug/blob/main/src/werkzeug/security.py. The function generate_password_hash is located on line 60. In this function it takes the plaintext string and computes the sha256 hash with a salt. The rest of the function is fairly straightforward. On line 87, the function gen_salt() is called which is located on line 19. This function takes an input length as an integer and computes a salt using the Python secrets library. It then returns this string and the _hash_internal function is called. This function is located on line 27 and it encodes the plain text password and encodes it into "utf-8" before using the Python hashlib library to compute and return the hex value of the hashed password. From there the return string is formatted and sent back to us to give us a salted hashed password to safely store in our database. We store the value directly given to us by this function in our database.

check_password_hash

Purpose

Replace this text with some that answers the following questions for the above tech:

- This tech takes both a hashed password that is similar to the one returned by the function generate_password_hash and compares it to a plaintext password.
- This tech is specifically used in our login() function. The check_password_hash is located on line 144 in app.py.

Magic

- This technology takes our inputted password when a user is trying to log in and compares it to the hashed password we have stored in our database.
- The code that does this tech is located on line 92 of the repository linked here: https://github.com/pallets/werkzeug/blob/main/src/werkzeug/security.py. This function takes a pwhash similar to the one that is returned by the generate_password_hash function. We stored the result of that directly in our database. From there we call the function with the login input. This password checking function takes the plaintext string and takes the salt from our stored password in the database to compare them. It uses the _hash_internal function located on line 27 to return the password from the plaintext input. The function check_password_hash digests the result from the _hash_internal function and uses the python hmac library to compare_digest the two results and return True if the passwords match and False if the password do not match.

secure_filename

Purpose

Replace this text with some that answers the following questions for the above tech:

- This function allows us to ensure that any files inputted by a user are safely stored without causing any issues whether malicious or intentional to our code.
- This tech is specifically used on line 189 in our app.py code is used for storing profile pictures to our server to ensure that no profile pictures are stored with the same name, with spaces, or with names similar to files already established.

Magic

- This technology takes the inputted plaintext string for the filename and encodes it into ascii ignoring any potential ascii errors. It then replaces any spaces with _ and checks to make sure that none of the files are in the os already. It then returns the new filename.
- The code is located on line 174 in the following repository https://github.com/pallets/werkzeug/blob/main/src/werkzeug/utils.py. It takes the inputted filename, converts it to unicodedata, then back to ascii and then removes any values that would not be suitable for a filename returning that value to us. This allows us to store filenames without worrying about them.