

# **Assignment 2**

## **Development Team Project: Coding Output**

Thien Liu & Neelam Pirbhai-Jetha

# Presentation Outline



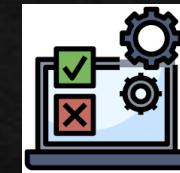
Introduction &  
Objective of this app



Demo



Security Features



Testing

## 1. INTRODUCTION

### USERS:

Main Admin

Create accounts+Full CRUD operations



Internal Users

Full CRUD operations of their documents+Read

External Users

Read Only

- ❖ Since security challenges (digital asset thefts and lack of protection of forensics data) and malevolent attacks and tampering of databases (Chopade & Pachghare, 2019) are numerous, our aim is to review Digitpol's system so that it follows the security best practices (OWASP, 2021; ISO/IEC 27000:2018).
- ❖ The main objective is to develop an application that provides a secure repository for Digipol that will allow authorised stakeholders to securely manage their documents in a protected repository which respects data privacy.
- ❖ The three aspects of software architecture – confidentiality, integrity, availability (CIA) – aided by authentication, authorisation, and non-reputability will be applied (Pillai, 2017).



# Flask Authorize

## Overview

Flask-Authorize is a Flask extension designed to simplify the process of incorporating Access Control Lists (ACLs) and Role-Based Access Control (RBAC) into applications housing sensitive data, allowing developers to focus on the actual code for their application instead of logic for enforcing permissions. It uses a unix-like permissions scheme for enforcing access permissions on existing content, and also provides mechanisms for globally enforcing permissions throughout an application.

There are quite a few packages designed to simplify the process of adding ACLs and RBAC to a Flask application:

- [Flask-Principal](#)
- [Flask-ACL](#)
- [Flask-RBAC](#)
- [Flask-Security](#)

And each provides a different developer experience and makes different assumptions in their design. This package is yet another take at solving the same problem, resulting in a slightly different development experience when working with Flask applications. The developers of this package recommend you check out these alternatives along with Flask-Authorize to see if they fit your needs better.

**Hierarchical authentication:  
flask-authorize, a library in flask to give different roles to users**

## About

Flask-Authorize provides utilities to help with user/role/group based app permissions and content CRUD authorization.

## Useful Links

[Flask-Authorize @ PyPI](#)  
[Flask-Authorize @ github](#)  
[Issue Tracker](#)

## Table of Contents

Flask-Authorize

- Overview
  - [A Minimal Application](#)
  - [Usage without Flask-Login](#)
- [User Guide](#)

# Running flask on the command prompt

- ❖ cd C:\Users\Neelam Pirbhai-Jetha\Desktop\thien\_neelam\_07february2022\ssd-secure-repository-develop
- ❖ flask run

```
C:\Users\Neelam Pirbhai-Jetha>cd C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop
C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop>flask run
* Environment: production
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Secure Repo

127.0.0.1:5000/auth/login?next=%2Fmain%2F

Home My Account Logout

Log In

Email

Password

Login

Go to `create_user.py` to get the login & password

internal\_created@awesomerepo.com  
Qwerty@123

127.0.0.1:5000/document/

Home My Account Logout

127.0.0.1:5000/auth/login?next=%2Fmain%2F

Home My Account Logout

Log In

Email

Password

Login

#	Document	Upload Document
1	<b>Title:</b> 2021-03-01_All_About_History.pdf <b>Created At:</b> Feb 05 2022 - 02:53 PM <b>Size:</b> 60.9 MB <b>Hash:</b> c974316944e94e869b66e31bb097dbd0b0993d692a0bfb04d8e616b6636a13a6 <a href="#">View</a> <a href="#">Download</a> <a href="#">Delete</a>	
2	<b>Title:</b> 7HabitHighlyE.fective_full.pdf <b>Created At:</b> Feb 05 2022 - 02:52 PM <b>Size:</b> 5.6 MB <b>Hash:</b> b330aa220d79ff400902772a6be2786f2a782cf67d7c7bc6a420d7a42245f65 <a href="#">View</a> <a href="#">Download</a> <a href="#">Delete</a>	
3	<b>Title:</b> Writing_Task_2_E-book_Oct_-_Jan_22.pdf <b>Created At:</b> Feb 05 2022 - 11:20 AM <b>Size:</b> 1.7 MB <b>Hash:</b> 151aeb6b5e2458da99308fb7fbbc59c198864a78241534abe4b467652a2b0552 <a href="#">View</a> <a href="#">Download</a> <a href="#">Delete</a>	

Home

My Account

Logout

## Profile

User ID: 3

Email: internal\_created@awesomerepo.com

Page appears on clicking  
on “My Account”

Home

My Account

Logout

## Log In

Email

Password

Login

Returns to login page  
when we click ‘logout’  
button

← → C ① 127.0.0.1:5000/document/

Home My Account Logout

My Documents

Upload Document

#	Document
1	<b>Title:</b> 2021-03-01_All_About_Myself.pdf <b>Created At:</b> Feb 05 2022 - 02:53 PM <b>Size:</b> 60.9 MB <b>Hash:</b> c974316944eb66e31bb097dbd0b0993d692a0bfb04d8e616b6636a13a6 View Download Delete
2	<b>Title:</b> 7HabitHighlyEffective_full.pdf <b>Created At:</b> Feb 05 2022 - 02:52 PM <b>Size:</b> 5.6 MB <b>Hash:</b> b330aa220d79ff400902772a6be2786f2a782cf67d7c7bc6a420d7a42245f65 View Download Delete
3	<b>Title:</b> Writing_Task_2_E-book_Oct_-Jan_22.pdf <b>Created At:</b> Feb 05 2022 - 11:20 AM <b>Size:</b> 1.7 MB <b>Hash:</b> 151aeb6b5e2458da99308fb7fbbc59c198864a78241534abe4b467652a2b0552 View Download Delete

Hash: check the integrity of the file (original file and uploaded file= same, not modified)

“Download” option, which was first implemented will have to be made more secure. Considering security challenges and sensitive data exposure - Enable download – encrypt – only on our tool ( but this has not yet created, and must be researched further. This was one of the Members of the Jury, Dr Buckley’s thought)

Research shows that most security issues arise from software vulnerabilities, especially in its “configuration, design, and implementation”, and the “lack of knowledge about security concerns” (Medeiros et al, 2017)

### 3. SECURITY

## Security features of our app

Encryption -  
bcrypt

Authentication – authorisation  
(validators)

Time constraints: other securities – adding fernet/cryptography

manage session timeout

## bcrypt

```
create_user.py > X
create_user.py > main
1 import email
2 from app import db, bcrypt
3 from app.auth.models import User
4 import sys
5
6 def main():
7     password = "Qwerty@123"
8     hashed_password = bcrypt.generate_password_hash(password)
9     user1 = User(
10         name = "Thien",
11         email = "thien@awesomerepo.com",
12         password = hashed_password,
13         role = 1,
14         status = 2
15     )
16     user2 = User(
17         name = "Neelam",
18         email = "neelam@awesomerepo.com",
19         password = hashed_password,
20         role = 1,
21         status = 2
22     )
23     user3 = User(
24         name = "Created Internal Staff",
25         email = "internal_created@awesomerepo.com",
26         password = hashed_password,
27         role = 2,
28         status = 1
29     )
30     db.session.add(user1)
31     db.session.add(user2)
32     db.session.add(user3)
```

session.permanent = True  
(sets after user successfully  
log in)

However, Dr Buckley, Member  
of the Jury suggested to  
reduce the session timeout  
(240 mins, being too long for  
the session)

```
config.py > T
1 import os
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31 config = {
```

```
9     CSRF_ENABLED = True
10    CSRF_SESSION_KEY = os.urandom(24)
11
12 # Session
13 PERMANENT_SESSION_LIFETIME = timedelta(minutes=240)
14
15 class DevelopmentConfig(Config):
16     SECRET_KEY = os.getenv('SECRET_KEY_DEV', os.urandom(24))
17     DATABASE_NAME = "awesome_repo_dev.db"
18     SQLALCHEMY_DATABASE_URI = 'sqlite:///+' + os.path.join(BASE_DIR, DATABASE_NAME)
19     ADMINDASHBOARD_VISIBLE = True
20
21 class TestingConfig(Config):
22     SECRET_KEY = os.getenv('SECRET_KEY_TEST', os.urandom(24))
23     DATABASE_NAME = "awesome_repo_test.db"
24     SQLALCHEMY_DATABASE_URI = 'sqlite:///+' + os.path.join(BASE_DIR, DATABASE_NAME)
25     ADMINDASHBOARD_VISIBLE = True
26
27 class ProductionConfig(Config):
28     SECRET_KEY = os.getenv('SECRET_KEY_PROD', os.urandom(24))
29     DATABASE_NAME = "awesome_repo_prod.db"
30     SQLALCHEMY_DATABASE_URI = 'sqlite:///+' + os.path.join(BASE_DIR, DATABASE_NAME)
31     ADMINDASHBOARD_VISIBLE = False
32
33 config = {
34     'development': DevelopmentConfig,
35     'testing': TestingConfig,
36     'production': ProductionConfig
37 }
```

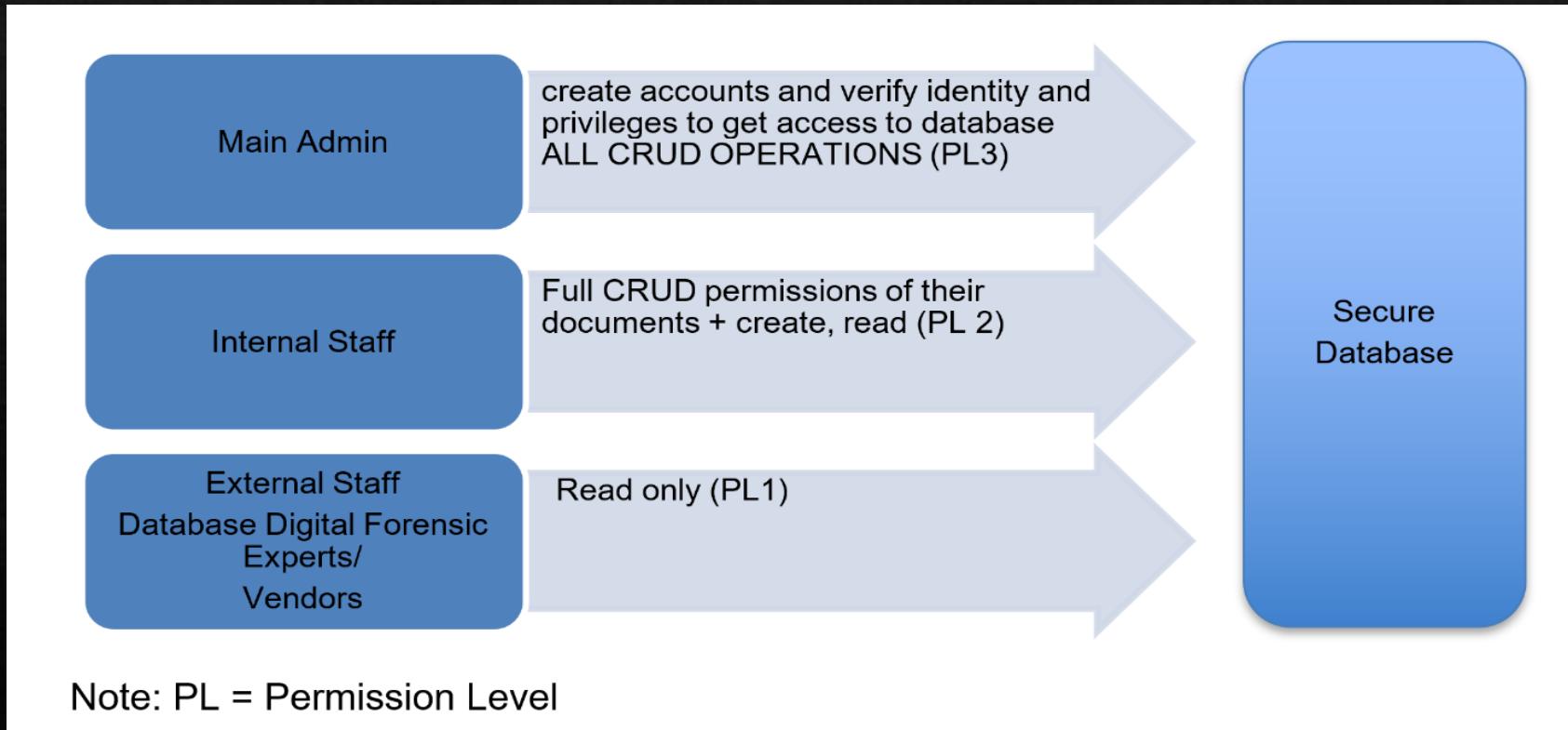
Three different environmental configurations 

The development/testing environment will not go public to the world outside. At the business level, a VPN connection will be required to access the resources from these environments

Modifications in the development config won't affect the others.

New features or issue fix – can be done in the development config, and then deployed to the testing config environment to verify if they pass the testing, then go public for the user (production config deployment)

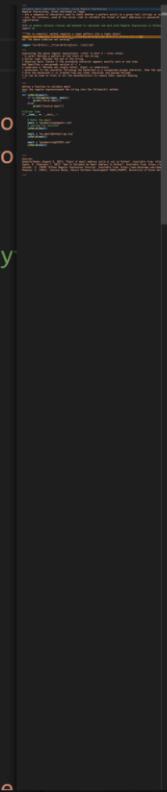
- ❖ Users do not register
- ❖ Admin creates and activates account
- ❖ Admin has full control of the system.
- ❖ Internal and external – have different permissions, but this has not been done yet



```

1 """
2     validate email addresses in Python, using Regular Expressions
3 Regular Expressions, often shortened as regex,
4 - are a sequence of characters used to check whether a pattern exists in a given text (string) o
5 - are, for instance, used at the server side to validate the format of email addresses or passwo
6 registration
7 """
8 #The re module contains classes and methods to represent and work with Regular Expressions in Py
9 import re
10
11 """The re.compile() method compiles a regex pattern into a regex object
12 regex = re.compile(r'([A-Za-z0-9]+[._-])*[A-Za-z0-9-]+@[A-Za-z0-9-]+\(\.[A-Z|a-z]{2,}\)+')
13 but the above codeline not working"""
14
15 regex='^@[a-z0-9]+[._-]?[\a-z0-9]+[@]\w+[. ]\w{2,3}$'
16
17 """
18 explaining the above regular expressions: (refer to Unit 4 - class notes)
19 ^ A caret: Matches a pattern at the start of the string
20 $ Dollar sign. Matches the end of the string.
21 ? Question mark: Checks if the preceding character appears exactly zero or one time.
22 And Specifies a non-greedy version of +, *
23 w Lowercase w: Matches any single letter, digit, or underscore.
24 \ backslash: If the character following the backslash is a recognized escape character, then the character then the
25 . Else the backslash () is treated like any other character and passed through.
26 . It can be used in front of all the metacharacters to remove their special meaning
27 """
28
29 """
30 define a function to validate email
31 pass the regular expressionand the string into the fullmatch() method
32 """
33 def isValid(email):
34     if re.fullmatch(regex, email):
35         print("Valid email")
36     else:
37         print("Invalid email")
38
39 # Driver Code
40 if __name__ == '__main__':
41
42     # Enter the email
43     email = "goodmorning1@gmail.com"
44     # calling run function
45     isValid(email)
46
47     valid_email = "goodmorning1@gmail.com"

```



# REGEX (Regular Expressions)

"""

There are various Python packages and APIs available to use instead of Regex:

- [email-validator](#)
- [pylsEmail](#)
- [py3-validate-email](#)

"""

We used  
wtforms.validators

```
39 # Driver Code
40 if __name__ == '__main__':
41
42     # Enter the email
43     email = "goodmorning1@gmail.com"
44     # calling run function
45     isValid(email)
46
47     email = "my.email@hotmail-go.org"
48     isValid(email)
49
50     email = "goodmorning622022.com"
51     isValid(email)
52
53
54 """
55 Sources:
56 GeeksforGeeks (August 8, 2021) "Check if email address valid or not in Python". Available from:
57 Gupta, R. (February 7, 2021) "How To Validate An Email Address In Python". Available from: https://www.geeksforgeeks.org/check-if-email-address-valid-or-not-in-python/
58 Jaiswal, S. (2020) Python Regular Expression Tutorial. Available from: https://www.datacamp.com/courses/python-regular-expressions-tutorial
59 Peoples, C. (2021), Lecture Notes, Secure Software Development SSDCS_PCOM7E, University of Essex
60
61
```

PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE

Python Debug Console + ▾ ^

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional> c:; cd 'c:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional' & 'C:\Users\Neelam Pirbhai-Jetha\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\Neelam Pirbhai-Jetha\.vscode\extensions\ms-python.python-2022.0.1786462952\pythonFiles\lib\python\debugpy\launcher' '59616' '--' 'c:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional\rex.py'
Valid email
Invalid email
Invalid email
```

Valid or not?

Some e-mail  
addresses that are  
being checked

# Screenshot of email validator of our app (instead of Regex)

The screenshot shows a code editor interface with a dark theme. On the left is the Explorer sidebar, which lists the project structure. The 'forms.py' file is selected in the sidebar, indicated by a blue highlight bar. The main area displays the code for 'forms.py'. The code defines two forms: 'LoginForm' and 'ChangePasswordForm'. The 'LoginForm' class contains fields for email and password with validation rules. The 'ChangePasswordForm' class contains fields for current password, new password, and confirm password. A custom validation method 'validate\_current\_password' is defined for the current password field, which checks if the user exists and if the provided password matches the hashed password stored in the database using bcrypt.

```
EXPLORER ... forms.py test_create_user.py

app > auth > forms.py > ...
1 from flask_wtf import FlaskForm, RecaptchaField
2 from wtforms import StringField, PasswordField, SubmitField
3 from wtforms.validators import DataRequired, Email, ValidationError
4 from flask_login import current_user
5 from app.errors.authErrors import (
6     InvalidLoginSessionError,
7     InvalidCurrentPasswordError,
8     PasswordMismatchError
9 )
10 from app.auth.models import User
11 from app import bcrypt

13 class LoginForm(FlaskForm):
14     email = StringField('Email',
15         validators=[DataRequired(), Email()])
16     password = PasswordField('Password', validators=[DataRequired()])
17     submit = SubmitField('Login')
18 class ChangePasswordForm(FlaskForm):
19     current_password = StringField('Current Password')
20     new_password = PasswordField('New Password')
21     confirm_password = PasswordField('Confirm Password')
22     submit = SubmitField('Update')

24     def validate_current_password(self, current_password):
25         if current_password == "" or current_password is not None:
26             if current_user.get_id() is not None:
27                 user = User.query.filter_by(id=current_user.get_id()).first()
28                 if not bcrypt.check_password_hash(user.password, current_password.data):
29                     raise InvalidCurrentPasswordError()
30             else:
31                 raise InvalidLoginSessionError()
```

```
config.py ×
config.py > 🐍 TestingConfig
1 import os
2 from datetime import timedelta
3
4 BASE_DIR = os.path.abspath(os.path.dirname(__file__))
5 class Config(object):
6     DEBUG = True
7     TESTING = False
8     SQLALCHEMY_TRACK_MODIFICATIONS = False
9     CSRF_ENABLED = True
10    CSRF_SESSION_KEY = os.urandom(24)
11
12 # Session
13 PERMANENT_SESSION_LIFETIME = timedelta(minutes=240)
14
15 class DevelopmentConfig(Config):
16     SECRET_KEY = os.getenv('SECRET_KEY_DEV', os.urandom(24))
17     DATABASE_NAME = "awesome_repo_dev.db"
18     SQLALCHEMY_DATABASE_URI = 'sqlite:///{}' + os.path.join(BASE_DIR, DATABASE_NAME)
19     ADMINDASHBOARD_VISIBLE = True
20
21 class TestingConfig(Config):
22     SECRET_KEY = os.getenv('SECRET_KEY_TEST', os.urandom(24))
23     DATABASE_NAME = "awesome_repo_test.db"
24     SQLALCHEMY_DATABASE_URI = 'sqlite:///{}' + os.path.join(BASE_DIR, DATABASE_NAME)
25     ADMINDASHBOARD_VISIBLE = True
26
27 class ProductionConfig(Config):
28     SECRET_KEY = os.getenv('SECRET_KEY_PROD', os.urandom(24))
29     DATABASE_NAME = "awesome_repo_prod.db"
30     SQLALCHEMY_DATABASE_URI = 'sqlite:///{}' + os.path.join(BASE_DIR, DATABASE_NAME)
31     ADMINDASHBOARD_VISIBLE = False
32
33 config = {
34     'development': DevelopmentConfig
```

GENERATING SECRET  
KEYS: random was used

not-secure-otp.py > ...

```
1 # Python Program for simple OTP genertaor
2
3 """
4 random is a module in Python that generates random numbers
5 However, being completely deterministic, it is not suitable for all purposes,
6 and is completely unsuitable for cryptographic purposes.
7
8 The pseudo-random generators of this module should not be used for security purposes.
9 For security or cryptographic uses, see the secrets module
10
11 """
12
13 import random as r
14 # function for otp generation (otpgen)
15 def otpgen():
16     otp=""
17     for i in range(6):
18         otp+=str(r.randint(1,9))
19     print ("Your One Time Password is " + otp)
20
21 otpgen()
22
23 """
24 Sources:
25 https://docs.python.org/3/library/random.html
26 """
```

PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE

Python Debug

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional> c;; cd 'c:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional' & 'C:\Users\Neelam Pirbhai-Jetha\AppData\Local\Programs\Python\Python39\python.exe' 'c:\Users\Neelam Pirbhai-Jetha\.vscode\extensions\ms-python.python-2022.0.1786462952\pythonFiles\lib\python\debugpy\launcher' '49696' '--' 'c:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\secure-otp.py'
```

Your One Time Password is 245463

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional> []
```

Random generator is not secure. Use 'secrets'

+ OTP has not been included yet in the Flask app

filesError.py X test\_create\_user.py

app > errors > filesError.py > FileAlreadyExistsError

```
1 class FileAlreadyExistsError(Exception):
2     def __init__(self, filename: str):
3         self.filename = filename
4         super().__init__()
5
6 class FileInsertionError(Exception):
7     def __init__(self, filename: str):
8         self.filename = filename
9         super().__init__()
10
11 class FileUpdateError(Exception):
12     def __init__(self, filename: str):
13         self.filename = filename
14         super().__init__()
15 class FileDeletionError(Exception):
16     def __init__(self, filename: str):
17         self.filename = filename
18         super().__init__()
```

app > errors > authErrors.py > InvalidLoginSessionError

```
1 class InvalidLoginSessionError(Exception):
2     def __init__(self):
3         super().__init__()
4
5 class InvalidCurrentPasswordError(Exception):
6     def __init__(self):
7         super().__init__()
8
9 class PasswordMismatchError(Exception):
10    def __init__(self):
11        super().__init__()
```

Codes written to raise errors if input of  
data is wrong

# Flask-Hashing

Flask-Hashing is a Flask extension that provides an easy way to hash data and check a hash of a value against a given hash. Flask-Hashing uses *hashlib* to actually hash data.

The main use case for hashing in web applications is password storage. Different choices are not password-specific.

Even if we use these libraries for our development phase, there might be some changes (in progress)

The image shows a file explorer on the left and a code editor on the right. The file explorer lists several Python files: models.py, document, \_\_pycache\_\_, \_\_init\_\_.py, controllers.py, forms.py, models.py, errors, \_\_pycache\_\_, \_\_init\_\_.py, authErrors.py, filesError.py, main, \_\_pycache\_\_, \_\_init\_\_.py, controllers.py, models.py, services, \_\_pycache\_\_, \_\_init\_\_.py, and FileService.py. The FileService.py file is selected and its content is displayed in the code editor.

```
1 from app import db
2 from app.document.models import File
3 from werkzeug.datastructures import FileStorage
4 from werkzeug.utils import secure_filename
5 from hashlib import sha256
6 from app.errors.filesError import (
7     FileAlreadyExistsError,
8     FileInsertionError
9 )
10
11
12 class FileService:
13
14     @classmethod
15     def get_user_files(cls, user_id: int):
16         files = db.session.query(File).filter_by(owner_id=user_id).order_by(
17             File.created_at.desc()
18         )
19
20         return files
21
22     @classmethod
23     def create_file(cls, uploaded_file: FileStorage, user_id: int):
24         filename = secure_filename(uploaded_file.filename)
25         if db.session.query(File).filter_by(owner_id=user_id, title=filename).first() is not None:
26             raise FileAlreadyExistsError(filename)
27         blob = uploaded_file.read()
28         # FileStorage class (which is the class to handle uploaded file in Flask)
29         # points to end of file after every action (saving or reading).
30         uploaded_file.stream.seek(0)
31         size = len(blob)
32         f_hash = sha256(blob).hexdigest()
```

#### 4. TESTING

- ❖ The software testing activity is very expensive, critical and complex, and a software without testing is most dangerous and leads to more expensiveness. Software testing is, therefore, an essential activity before the release of any software (Madhavi, 2016; Syaikhuddin et al, 2018).
- ❖ We have tried to apply some of the tools we have learnt in this class (e.g pylint, flake8, mccabe, cyclomatic complexity)

- ❖ error/exception handling: manually (wrong password, wrong email)
- ❖ This is part of the ERROR HANDLER: if our system cannot find the route, it will redirect to the 404.html template

```
SSD-SECURE-REPOSITORY-... app > _init_.py > index
> __pycache__ py
> .vscode
app
> __pycache__
> admin
> auth
> document
> errors
> main
> services
> templates
> _init_.py
> test_cases
> venv
.gitignore
awesome_repo_dev.db
awesome_repo_prod....
config.py
create_user.py
Diffile

32     app.app_context().push()
33
34     # Import a module / component using its blueprint handler variable (mod_auth)
35     from app.auth.controllers import auth as auth_module
36     from app.main.controllers import main as main_module
37     from app.admin.controllers import admin as admin_module
38     from app.document.controllers import document as document_module
39
40     @app.errorhandler(404)
41     def not_found(error):
42         return render_template('404.html'), 404
43
44     @app.route('/')
45     def index():
46         return redirect(url_for('main.home'))
47
48     # Register blueprint(s)
49     # app.register_blueprint(mod_auth)
50     # app.register_blueprint(mod_main)
51     # app.register_blueprint(mod_admin)
52     # app.register_blueprint(mod_document)
```

← → C ⓘ 127.0.0.1:5000/auth/login?next=%2Fmain%2F

Home My Account Logout

## Log In

Email

Invalid email address.

Password

Login

Another Demo on error-handling on our website: invalid email address

← → C ⓘ 127.0.0.1:5000/document/upload

Home My Account Logout

Select a file to upload

Choose File No file chosen

Submit

! Please select a file.

Message pops up as a file must be selected to be uploaded

# Testing: pylint

```
secure_otp.py > ...
1  """
2  Generate a secure secret code
3  Instead of random, use secrets module
4
5  """
6  import secrets
7  # Getting systemRandom class instance out of secrets module
8
9  username = input("Enter Username: ")
10
11 otp = secrets.token_hex(16)
12 print("Hello " + username + "!" + " Your OTP is " + otp)
13
14 password=input("Enter the OTP:")
15 if password==otp:
16     print("OTP is correct.")
17 else:
18     print("OTP is incorrect.")
19
20
21
```

PROBLEMS    OUTPUT    **TERMINAL**    SQL CONSOLE    DEBUG CONSOLE

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional> pylint secure_otp.py
*****
Module secure_otp
secure_otp.py:2:29: C0303: Trailing whitespace (trailing-whitespace)
secure_otp.py:21:0: C0305: Trailing newlines (trailing-newlines)
```

```
-----  
Your code has been rated at 7.50/10 (previous run: 7.50/10, +0.00)
```

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\ASSIGNMENT2\codes_additional> █
```

OTP will be  
added later to  
our codes

In the Terminal: pylint file\_name.py

# Pylint testing on config.py

PROBLEMS    OUTPUT    TERMINAL    SQL CONSOLE    DEBUG CONSOLE

[powershell] + ▾

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> pylint config.py
*****
Module config
config.py:11:0: C0303: Trailing whitespace (trailing-whitespace)
config.py:35:0: C0304: Final newline missing (missing-final-newline)
config.py:1:0: C0114: Missing module docstring (missing-module-docstring)
config.py:5:0: C0115: Missing class docstring (missing-class-docstring)
config.py:5:0: R0205: Class 'Config' inherits from object, can be safely removed from bases in python3 (useless-object-inheritance)
config.py:5:0: R0903: Too few public methods (0/2) (too-few-public-methods)
config.py:15:0: C0115: Missing class docstring (missing-class-docstring)
config.py:15:0: R0903: Too few public methods (0/2) (too-few-public-methods)
config.py:20:0: C0115: Missing class docstring (missing-class-docstring)
config.py:20:0: R0903: Too few public methods (0/2) (too-few-public-methods)
config.py:25:0: C0115: Missing class docstring (missing-class-docstring)
config.py:25:0: R0903: Too few public methods (0/2) (too-few-public-methods)
```

-----

Your code has been rated at 5.38/10

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure
```

## What is Pylint?

Pylint is a tool that checks for errors in Python code, tries to enforce a coding standard, and looks for code smells. It can also look for certain type errors, it can recommend suggestions about how particular blocks can be refactored and can offer you details about the code's complexity.

Other similar projects would include [pychecker](#) (now defunct), [pyflakes](#), [flake8](#), and [mypy](#). The default coding style used by Pylint is close to [PEP 8](#).

Pylint will display a number of messages as it analyzes the code and it can also be used for displaying some statistics about the number of warnings and errors found in different files. The messages are classified under various categories such as errors and warnings.

Last but not least, the code is given an overall mark, based on the number and severity of the warnings and errors.

# Pylint testing on create\_user.py

PROBLEMS    OUTPUT    TERMINAL    SQL CONSOLE    DEBUG CONSOLE

powershell    +        ^    X

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> pylint create_user.py
*****
Module create_user
create_user.py:11:40: C0303: Trailing whitespace (trailing-whitespace)
create_user.py:13:17: C0303: Trailing whitespace (trailing-whitespace)
create_user.py:18:41: C0303: Trailing whitespace (trailing-whitespace)
create_user.py:20:17: C0303: Trailing whitespace (trailing-whitespace)
create_user.py:25:51: C0303: Trailing whitespace (trailing-whitespace)
create_user.py:27:17: C0303: Trailing whitespace (trailing-whitespace)
create_user.py:1:0: C0114: Missing module docstring (missing-module-docstring)
create_user.py:6:0: C0116: Missing function or method docstring (missing-function-docstring)
create_user.py:30:4: E1101: Instance of 'scoped_session' has no 'add' member (no-member)
create_user.py:31:4: E1101: Instance of 'scoped_session' has no 'add' member (no-member)
create_user.py:32:4: E1101: Instance of 'scoped_session' has no 'add' member (no-member)
create_user.py:33:4: E1101: Instance of 'scoped_session' has no 'commit' member (no-member)
create_user.py:1:0: W0611: Unused import email (unused-import)
create_user.py:4:0: C0411: standard import "import sys" should be placed before "from app import db, bcrypt" (wrong-import-order)
```

-----  
Your code has been rated at -8.75/10

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> █
```

# Pylint testing on run.py

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> pylint run.py
***** Module run
run.py:3:0: C0304: Final newline missing (missing-final-newline)
run.py:1:0: C0114: Missing module docstring (missing-module-docstring)

-----
Your code has been rated at 0.00/10

PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop>
```

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> flake8 config.py
config.py:5:1: E302 expected 2 blank lines, found 0
config.py:11:1: W293 blank line contains whitespace
config.py:13:1: E305 expected 2 blank lines after class or function definition, found 1
config.py:15:1: E302 expected 2 blank lines, found 1
config.py:18:80: E501 line too long (82 > 79 characters)
config.py:20:1: E302 expected 2 blank lines, found 0
config.py:23:80: E501 line too long (82 > 79 characters)
config.py:25:1: E302 expected 2 blank lines, found 0
config.py:28:80: E501 line too long (82 > 79 characters)
config.py:31:1: E305 expected 2 blank lines after class or function definition, found 1
config.py:35:2: W292 no newline at end of file
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop>
```

## Testing flake8

“Flake8 = Python library to check the code base against coding style (PEP8), programming errors (like “library imported but unused” and “Undefined name”) and to check cyclomatic complexity” [Freitas, V. “How to use Flake8”.

<https://simpleisbetterthancomplex.com/packages/2016/08/05/flake8.html>]

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> flake8 create_user.py
create_user.py:1:1: F401 'email' imported but unused
create_user.py:6:1: E302 expected 2 blank lines, found 1
create_user.py:10:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:10:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:11:14: E251 unexpected spaces around keyword / parameter equals
create_user.py:11:16: E251 unexpected spaces around keyword / parameter equals
create_user.py:11:41: W291 trailing whitespace
create_user.py:12:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:12:19: E251 unexpected spaces around keyword / parameter equals
create_user.py:13:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:13:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:13:18: W291 trailing whitespace
create_user.py:14:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:14:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:17:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:17:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:18:14: E251 unexpected spaces around keyword / parameter equals
create_user.py:18:16: E251 unexpected spaces around keyword / parameter equals
create_user.py:18:42: W291 trailing whitespace
create_user.py:19:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:19:19: E251 unexpected spaces around keyword / parameter equals
create_user.py:20:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:20:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:20:18: W291 trailing whitespace
create_user.py:21:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:21:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:24:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:24:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:25:14: E251 unexpected spaces around keyword / parameter equals
create_user.py:25:16: E251 unexpected spaces around keyword / parameter equals
create_user.py:25:52: W291 trailing whitespace
create_user.py:26:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:26:19: E251 unexpected spaces around keyword / parameter equals
create_user.py:27:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:27:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:27:18: W291 trailing whitespace
create_user.py:28:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:28:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:35:1: E305 expected 2 blank lines after class or function definition, found 1
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop>
```

# Testing mccabe

## flake8 --max-complexity 10 name\_of\_file.py

PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE

[powershell] + □ × ^

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> flake8 --max-complexity 10 create_user.py
create_user.py:1:1: F401 'email' imported but unused
create_user.py:6:1: E302 expected 2 blank lines, found 1
create_user.py:10:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:10:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:11:14: E251 unexpected spaces around keyword / parameter equals
create_user.py:11:16: E251 unexpected spaces around keyword / parameter equals
create_user.py:11:41: W291 trailing whitespace
create_user.py:12:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:12:19: E251 unexpected spaces around keyword / parameter equals
create_user.py:13:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:13:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:13:18: W291 trailing whitespace
create_user.py:14:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:14:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:17:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:17:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:18:14: E251 unexpected spaces around keyword / parameter equals
create_user.py:18:16: E251 unexpected spaces around keyword / parameter equals
create_user.py:18:42: W291 trailing whitespace
create_user.py:19:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:19:19: E251 unexpected spaces around keyword / parameter equals
create_user.py:20:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:20:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:20:18: W291 trailing whitespace
create_user.py:21:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:21:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:24:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:24:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:25:14: E251 unexpected spaces around keyword / parameter equals
create_user.py:25:16: E251 unexpected spaces around keyword / parameter equals
create_user.py:25:52: W291 trailing whitespace
create_user.py:26:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:26:19: E251 unexpected spaces around keyword / parameter equals
create_user.py:27:13: E251 unexpected spaces around keyword / parameter equals
create_user.py:27:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:27:18: W291 trailing whitespace
create_user.py:28:15: E251 unexpected spaces around keyword / parameter equals
create_user.py:28:17: E251 unexpected spaces around keyword / parameter equals
create_user.py:35:1: E305 expected 2 blank lines after class or function definition, found 1
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop>
```

PROBLEMS OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE

[powershell] + □ ×

```
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop> flake8 --max-complexity 10 config.py
config.py:5:1: E302 expected 2 blank lines, found 0
config.py:11:1: W293 blank line contains whitespace
config.py:13:1: E305 expected 2 blank lines after class or function definition, found 1
config.py:15:1: E302 expected 2 blank lines, found 1
config.py:18:80: E501 line too long (82 > 79 characters)
config.py:20:1: E302 expected 2 blank lines, found 0
config.py:23:80: E501 line too long (82 > 79 characters)
config.py:25:1: E302 expected 2 blank lines, found 0
config.py:28:80: E501 line too long (82 > 79 characters)
config.py:31:1: E305 expected 2 blank lines after class or function definition, found 1
config.py:35:2: W292 no newline at end of file
PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien_neelam_07february2022\ssd-secure-repository-develop>
```

# Cyclomatic complexity

- ❖ Radon is a Python tool that computes various metrics from the source code. Radon can compute:
  - **McCabe's complexity**, i.e. cyclomatic complexity
  - **raw** metrics (these include SLOC, comment lines, blank lines, &c.)
  - **Halstead** metrics (all of them)
  - **Maintainability Index** (the one used in Visual Studio)
  - Source: <https://pypi.org/project/radon/>
  - In cmd: pip install radon

PROBLEMS

OUTPUT

TERMINAL

SQL CONSOLE

DEBUG CONSOLE

powershell + ▾

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien\_neelam\_07february2022\ssd-secure-repository-develop> **radon cc config.py**

config.py

C 5:0 Config - A

C 15:0 DevelopmentConfig - A

C 20:0 TestingConfig - A

C 25:0 ProductionConfig - A

PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien\_neelam\_07february2022\ssd-secure-repository-develop> **radon mi config.py**

config.py - A

PS C:\Users\Neelam Pirbhai-Jetha\Desktop\thien\_neelam\_07february2022\ssd-secure-repository-develop> █

## The **cc** command

This command analyzes Python source files and compute Cyclomatic Complexity. The output can be filtered by specifying the **-n** and **-x** flags. By default, the complexity score is not displayed, the option **-s** (show complexity) toggles this behaviour. File or directories exclusion is supported through glob patterns. Every positional argument is interpreted as a path. The program then walks through its children and analyzes Python files. Every block will be ranked from A (best complexity score) to F (worst one). Ranks corresponds to complexity scores as follows:

CC score	Rank	Risk
1 - 5	A	low - simple block
6 - 10	B	low - well structured and stable block
11 - 20	C	moderate - slightly complex block
21 - 30	D	more than moderate - more complex block
31 - 40	E	high - complex block, alarming
41+	F	very high - error-prone, unstable block

## The **mi** command

This command analyzes Python source code files and compute the Maintainability Index score. Every positional argument is treated as a starting point from which to walk looking for Python files (as in the **cc** command). Paths can be excluded with the **-e** option. The Maintainability Index is always in the range 0-100. MI is ranked as follows:

MI score	Rank	Maintainability
100 - 20	A	Very high
19 - 10	B	Medium
9 - 0	C	Extremely low

Options

# Pip install pytest

- ❖ **PyTest** is a testing framework that allows users to write test codes using Python programming language. It helps to write simple and scalable test cases for databases, APIs, or UI. PyTest is mainly used for writing tests for APIs. It helps to write tests from simple unit tests to complex functional tests.
- ❖ **Which is better – pytest or unittest?**

Although both the frameworks are great for performing testing in python, pytest is easier to work with. The code in pytest is simple, compact, and efficient.

For unittest, we will have to import modules, create a class and define the testing functions within that class. [<https://www.pythonpool.com/python-unittest-vs-pytest/#:~:text=Which%20is%20better%20%E2%80%93%20pytest%20or,testing%20functions%20within%20that%20class.>]
- ❖ Create test\_filename.py
- ❖ In the Terminal,
- ❖ py.test filename.py
- ❖ Check <https://www.youtube.com/watch?v=OcD52IXq0e8>

# Equivalence Partitioning Testing or Equivalence Class Partitioning = black box testing

- ❖ To test, for instance, if the right number of digits have been entered (for the OTP).
  - ❖ Find the code for equivalence Testing

# Summing up

- ❖ We are thinking about
  - ❖ a dashboard of threat monitoring, a system audit logs to track the user's activities by looking at the logs (detect unusual activities)
  - ❖ Cryptography (fernet,...)