

WEB APPLICATION WITH PYTHON

NETWORK SECURITY / WEB SECURITY

CS 750 PYTHON
Loyola University
Maryland

HUTCHAGANIT PUTIPRAWAN
(HANNA)

HISTORICAL PERSPECTIVE

- **Major Topic:** Network Security / Web Security
- This web server project provides basics functions of online blog with some securities
- Basic security methods: random secret key, base64, Hashing with SHA-1, SHA-3

PROJECT REQUIREMENTS

- Python 2.7.9
- MySQL Server
- Virtual Environment (venv)
- Editor
- Internet Browser

TOOLS

- Sublime Text
- Terminal
- Chrome
- MySQLWorkbench

ALL FRAMEWORKS IN THIS PROJECT

- cffi==0.9.2
- cryptography==0.8.2**
- enum34==1.0.4
- Flask==0.10.1
- Flask-Admin==1.1.0
- Flask-Bcrypt==0.6.2
- Flask-Login==0.2.11**
- Flask-MySQL==1.2
- flask-paginate==0.2.8**
- Flask-SQLAlchemy==2.0
- Flask-Testing==0.4.2**
- Flask-WTF==0.11**
- gnureadline==6.3.3
- ipython==2.4.1
- itsdangerous==0.24
- Jinja2==2.7.3
- MarkupSafe==0.23
- MySQL-python==1.2.5
- pyasn1==0.1.7
- pycparser==2.12
- pycrypto==2.6.1**
- pysha3==0.3
- python-bcrypt==0.3.1
- six==1.9.0
- SQLAlchemy==0.9.8
- Werkzeug==0.10.1
- WTForms==2.0.2

* Only orange color that I actually experience
** Allow pip install

ORGANIZE FILES WITH BLUEPRINT

- Make a file to be a blueprint (create a path)

```
posts_blueprint = Blueprint(  
    'PostsFiles', __name__,  
    template_folder='templates'  
)
```

- Import the blueprint in `__init__.py`

```
from App.PostsFiles.views import posts_blueprint
```

- Register the blueprint in `__init__.py`

```
app.register_blueprint(posts_blueprint)
```

- Use the blueprint

```
@posts_blueprint.route('/showAllPosts<int:page>')
```

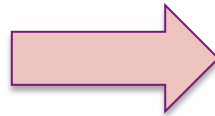
OVERALL ORGANIZATION OF THIS APP

Python_Web_1

- App << ALL Files
- venv << Virtual Environment for running the server
- run.py << Main for launching the application
- config.py << Set all configuration

App

- HomeFiles
- LogFiles
- PostsFiles
- UsersFiles
- static
- templates
- __init__.py
- models.py << ALL models for database
- share.py



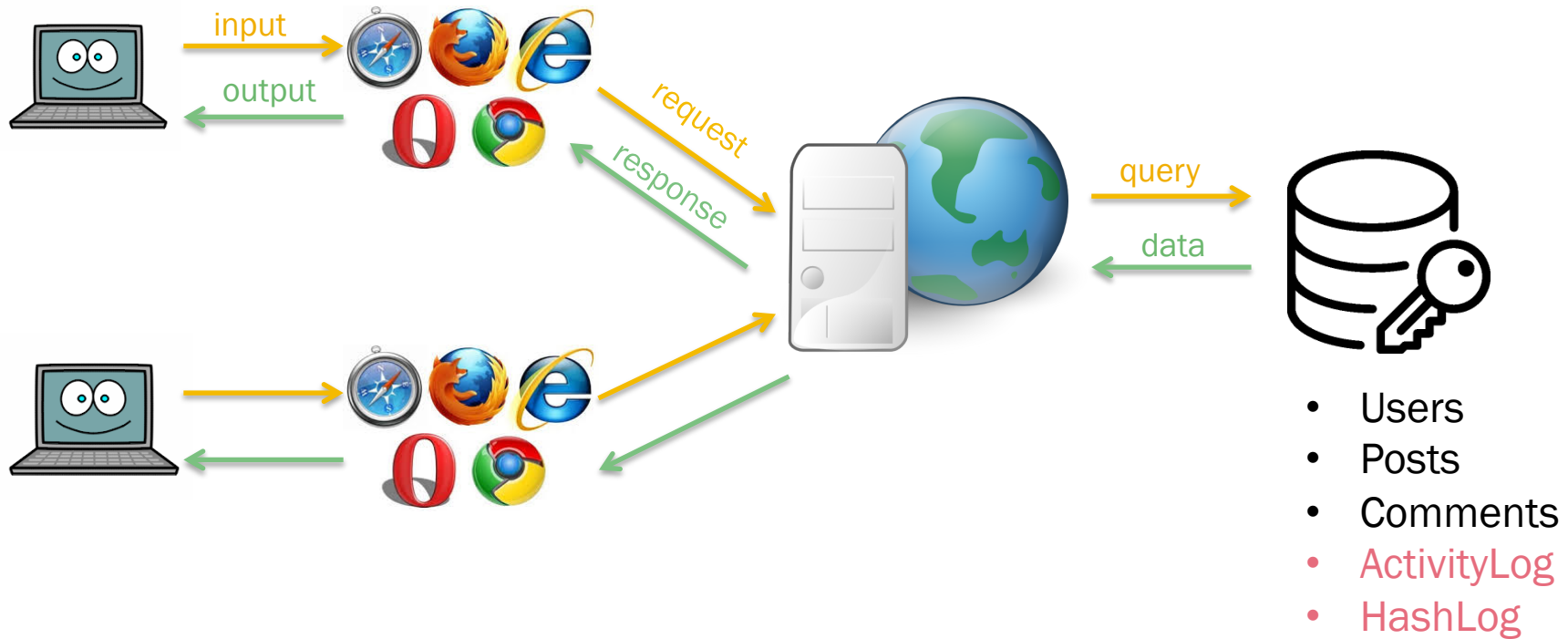
PostsFiles

- Templates << html files
- forms.py
- views.py

HOW TO RUN THIS WEB

- Start up the database server by using command line
 - `sudo mysql.server start`
- Activate the virtual environment
 - `source venv/bin/activate`
- Run the python file with command line
 - `python run.py` << run.py is the app name

FLOW DIAGRAM OF INPUT-PROCESS-OUTPUT (IPO)



DATABASES

Users

<u>user_id</u> (int)	user_name (varchar(50))	user_password (varchar(50))	user_lastLogin (datetime)
-------------------------	----------------------------	--------------------------------	------------------------------

Posts

<u>post_id</u> (int)	post_type (varchar(100))	post_title (varchar(100))	post_time (datetime)	post_privacy (varchar(10))	post_content (longtext)	post_authorID (int)
-------------------------	-----------------------------	------------------------------	-------------------------	-------------------------------	----------------------------	------------------------

Comments

<u>comment_id</u> (int)	comment_content (text)	comment_time (datetime)	comment_owner (int)	comment_postID (int)
----------------------------	---------------------------	----------------------------	------------------------	-------------------------

ActivityLog

<u>log_id</u> (int)	log_userID (int)	log_username (varchar(50))	log_ipAddr (varchar(15))	log_action (varchar(50))	log_time (datetime)	log_postID (int)	log_postTitle (varchar(100))	log_postType (varchar(100))	log_postPrivacy (varchar(10))	log_content (longText)
------------------------	---------------------	-------------------------------	-----------------------------	-----------------------------	------------------------	---------------------	---------------------------------	--------------------------------	----------------------------------	---------------------------

HashLog

<u>hash_id</u> (int)	hash_name (varchar(50))	hash_date (datetime)	hash_org (text)	hash_comp (text)	hash_isHack (varchar(10))
-------------------------	----------------------------	-------------------------	--------------------	---------------------	------------------------------

HASHING USERS' PASSWORD WITH WERKZEUG

- SHA-1
- Generate hash:
`generate_password_hash(user_password)`
- Compare hash:
`check_password_hash(OriginalHashPwd, ComparedHashPwd)`

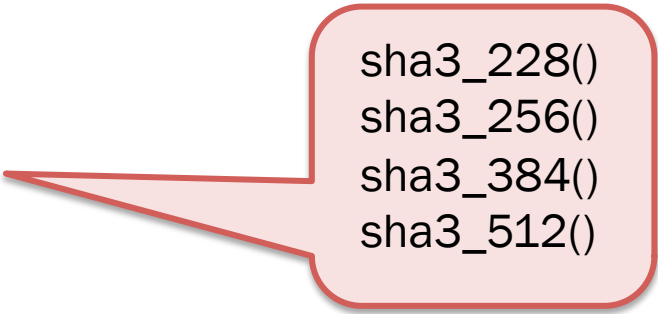
CHECK IF THE DATABASE IS MODIFIED WITH SHA-3

- The data has been modified since last login
- Modify database without passing through the web page

(Modify directly to the database)

- **SHA-3: Checking the hash**

```
def makeHash(data):  
    s = hashlib.sha3_512()  
    s.update(data)  
    return s.hexdigest()
```



```
sha3_228()  
sha3_256()  
sha3_384()  
sha3_512()
```

FLASK-SQLALCHEMY

db.Model VS Declarative base

db.Model

```
posts = db.session.query(Users.user_name, Posts.post_title, Posts.post_type,  
Posts.post_time, Posts.post_privacy).filter(Posts.post_authorID == Users.user_id).count()
```

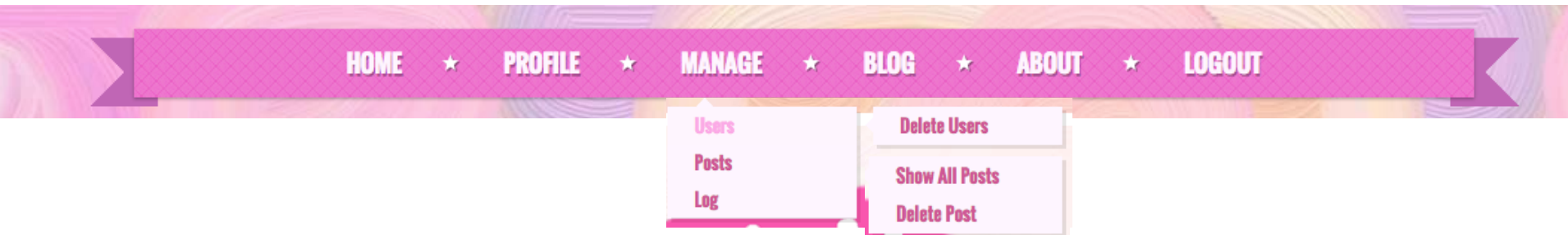
Declarative

```
posts = Posts.query.add_columns(Users.user_name, Posts.post_id, Posts.post_title,  
Posts.post_type, Posts.post_time, Posts.post_privacy).filter(Posts.post_authorID ==  
Users.user_id).count()
```

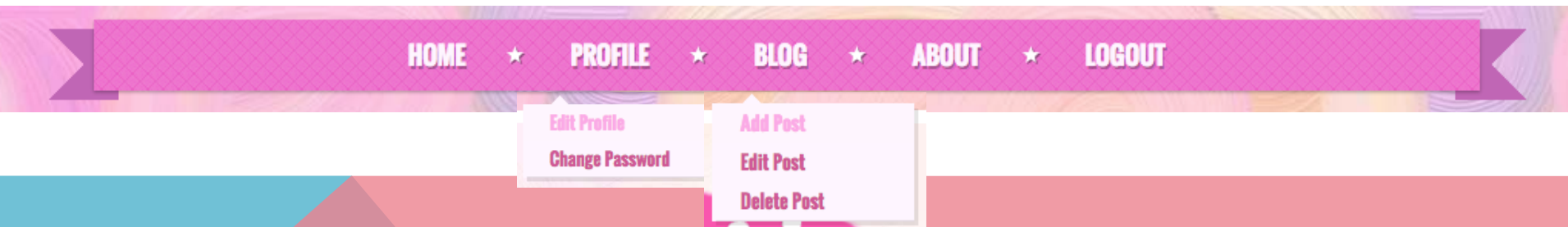
*Test query with ipython

SAMPLE OF INPUT/OUTPUT DATA

Login with an admin account



Login with a regular user



SAMPLE OF INPUT DATA


Adding a post


HOME ★ PROFILE ★ BLOG ★ ABOUT ★ LOGOUT

MAKE A POST!!

Post Type: Recipe - Desserts

Post Title: Carrot Cake Cupcakes

Content: 

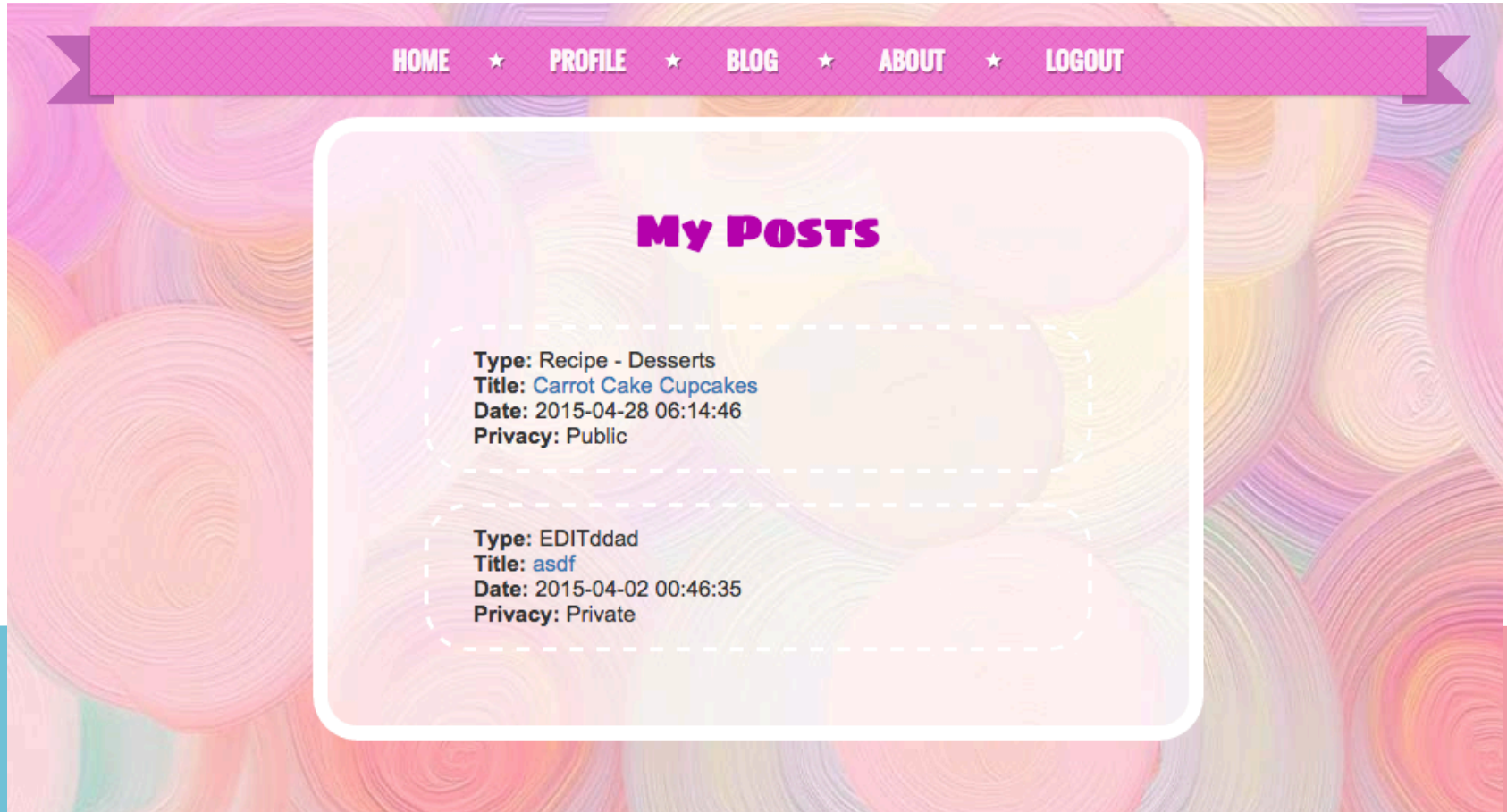
Privacy: Public 

Ingredients:

- 2 cups of cake flour
- 2 cups of granulated sugar 2 tsp. of baking soda
- 2 tsp. of cinnamon
- 1/2 tsp. of ginger
- 2 Tbsp. of vegetable oil
- 2 cups of grated carrots
- 4 eggs

SAMPLE OF OUTPUT DATA

Adding a post



SAMPLE OF INPUT DATA

Making a comment

[HOME](#) * [PROFILE](#) * [BLOG](#) * [ABOUT](#) * [LOGOUT](#)

RED VELVET CUPCAKES

Recipe - Desserts

Date: 2015-04-28 06:12:00

Ingredients:

- 1 box of German chocolate cake mix
- 1 cup of water
- 4 eggs
- 1/2 cup of sunflower oil
- 2 Tbsp. of unsweetened baking coco
- 2 Tbsp. of red food coloring

Instructions:

Preheat oven to 350F. Beat cake mix, eggs, water, oil, coco and food coloring together at a medium speed for 2 minutes. Fill 24 baking cups approximately 2/3 full and bake for 20 minutes or until a toothpick inserted in the center of a cupcake comes out clean. Allow for them to completely cool down.





Frosting Ingredients:

- 16 oz. of cream cheese, softened
- 9 cups of powdered sugar
- 1/3 cup of margarine
- 1 Tbsp. of vanilla
- 2 Tbsp. of milk

Instructions:

Beat cream cheese, margarine and vanilla on high speed until fluffy. Gradually add powdered sugar, vanilla and milk and beat until frosting is smooth.

ALL COMMENTS:

Comment:    

Looks yummy!! :)

Comment

SAMPLE OF OUTPUT DATA

Making a comment

RED VELVET CUPCAKES

Recipe - Desserts

Date: 2015-04-28 06:12:00

Ingredients:

- 1 box of German chocolate cake mix 1 cup of water
- 4 eggs
- 1/2 cup of sunflower oil
- 2 Tbsp. of unsweetened baking coco
- 2 Tbsp. of red food coloring

Instructions:
Preheat oven to 350F. Beat cake mix, eggs, water, oil, coco and food coloring together at a medium speed for 2 minutes. Fill 24 baking cups approximately 2/3 full and bake for 20 minutes or until a toothpick inserted in the center of a cupcake comes out clean. Allow for them to completely cool down.

Frosting Ingredients:

- 16 oz. of cream cheese, softened 9 cups of powdered sugar
- 1/3 cup of margarine
- 1 Tbsp. of vanilla
- 2 Tbsp. of milk

Instructions:
Beat cream cheese, margarine and vanilla on high speed until fluffy. Gradually add powdered sugar, vanilla and milk and beat until frosting is smooth.

ALL COMMENTS:

hputiprawan2@loyola.edu 2015-04-28 06:28:25

Looks yummy!! :)

Comment:

Comment

EXPECTATION/LEARNING

- Basic Python
- Database
- Networking and Security
- HTLM
- CSS
- Java Script

To-do List (Future Plan)

- Error + Stack trace page
- More functions after the database get hacked
- Image database
- Fix some layout/page

REFERENCES (1)

Virtual Environment:

- <http://docs.python-guide.org/en/latest/dev/virtualenvs/>

Partial Code + Learning Basic Python Tutorial:

- <https://www.youtube.com/playlist?list=PLLjmbh6XPGK4ISY747FUHXEI9IBxre4mM>
- <http://www.techillumination.in/2014/01/python-web-application-development.html>
- <https://www.youtube.com/playlist?list=PL0DA14EB3618A3507>

Random Secret Key:

- <http://flask.pocoo.org/docs/0.10/quickstart/>
- <https://gist.github.com/geoffalday/2021517>

Base64 Encode/Decode:

- <https://www.safaribooksonline.com/library/view/python-cookbook-3rd/9781449357337/ch06s10.html>

Hashing Password:

- <http://code.tutsplus.com/tutorials/intro-to-flask-signing-in-and-out-net-29982>

REFERENCES (2)

Flask-SQLAlchemy:

- <https://realpython.com/blog/python/using-flask-login-for-user-management-with-flask>
- <http://pythonhosted.org/Flask-SQLAlchemy/queries.html>
- http://docs.sqlalchemy.org/en/rel_0_9/orm/tutorial.html#querying-with-joins
- <http://stackoverflow.com/questions/270879/efficiently-updating-database-using-sqlalchemy-orm>
- http://www.reddit.com/r/flask/comments/2olr3s/af_how_on_earth_do_i_delete_a_row_with_sql_alchemy/
- <http://stackoverflow.com/questions/4186062/sqlalchemy-order-by-descending>
- http://w3facility.org/question/what-is-the-difference-between-the-declarative_base-and-db-model/
- <http://chris.improbable.org/2011/7/20/armin-ronacher-sqlalchemy-and-you/>

REFERENCE (3)

HTML/CSS/Jinja:

- <http://stackoverflow.com/questions/3206344/passing-html-to-template-using-flask-jinja2>
- <http://xing.github.io/wysihtml5/>
- <http://designmodo.com/retro-navigation-menu-css3/>
- <https://computerservices.temple.edu/creating-tables-html>
- <https://css-tricks.com/left-align-and-right-align-text-on-the-same-line>
- <http://stackoverflow.com/questions/8814472/how-to-make-an-html-back-link>

WTForms:

- <http://stackoverflow.com/questions/17192595/flask-wtf-selectfield-dynamic-input-and-validation>

Edit Database:

- <https://groups.google.com/forum/#!topic/wtforms/kmVQZOwpJYA>
- <http://stackoverflow.com/questions/7461962/mysql-how-to-store-aes-encrypted-data>

REFERENCE (4)

Trigger Database:

- http://www.techonthenet.com/mysql/triggers/after_update.php
- <https://www.youtube.com/watch?v=BeG4IHFjqVg>
- <http://stackoverflow.com/questions/7595714/how-to-write-a-trigger-to-abort-delete-in-mysql>

Timeout Login session:

- <http://runnable.com/UiIEmfuGwAlQAAAI/how-to-timeout-expire-a-flask-session-for-python>

Pagination:

- <http://flask-paginate.readthedocs.org/en/latest/>
- <http://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-ix-pagination>
- <http://stackoverflow.com/questions/25032744/how-to-return-multiple-models-tables-flask-sqlalchemy>

REFERENCE (5)

Get IP address:

- <http://stackoverflow.com/questions/3759981/get-ip-address-of-visitors-using-python-flask>

Pysha3:

- <https://pypi.python.org/pypi/pysha3/>
- http://www.mit.edu/~puzzle/2014/puzzle-solution/puzzle_with_answer_cronin/puzzle.py

Cryptography

- <http://docs.python-guide.org/en/latest/scenarios/crypto/>
- <https://www.youtube.com/watch?v=lsflaKpeB7Q>
- <http://www.tylerlesmann.com/2008/dec/19/encrypting-database-data-django/>