Topic	Phishing -1	
Class Description	**Students will be able to understand phishing and how it usually occurs, they will learn the entire process for phishing**	
Class	**C-224**	
Class time	**45 mins**	
Goal	• Understand about Phishing • Get & Post on Flask • Will create their own phishing design page	

Resources Required	• Teacher Resources:
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Visual Studio Code
	• Student Resources:
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Visual Studio Code	

Class structure	**Warm-Up** **Teacher - led Activity 1** **Student - led Activity 1** **Wrap-Up**	**10 mins** **10 mins** **20 mins** **5 mins**
WARM UP SESSION - 10mins		
Teacher Action		**Student Action**

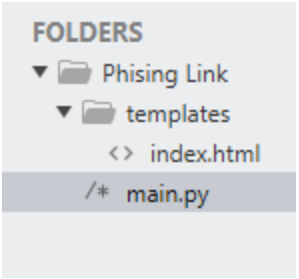
Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today?	**ESR**: Hi, thanks, yes, I am excited about it!
Q&A Session	
Question	**Answer**
Why do we use flask? A. For web framework and web applications B. To remove data from webpages C. To access data D. None of the above	**A**
What can we do with phishing links? A. Get user password B. Get user information C. Get username D. All of the above	**D**
TEACHER-LED ACTIVITY - 10mins	
Teacher Initiates Screen Share	
ACTIVITY • **Socket & Client Connection** • **Get screen size** • **Call the function**	
Teacher Action	**Student Action**
Okay, so you remember what we did in the last session? Great! Any doubts from last session?	

The teacher clarifies doubts (if any) Any idea what we will be doing in today's session?	
Remember, in the last class I got your username and password It's a kind of cyber attack that I've performed against you by stealing your username and password, this type of attacks known as phishing attacks Every day, we must have come across a number of phishing links Have you noticed how they create those phishing links, and most of the time users like us fall victim to them Can you share your experience about these phishing links Mostly in phishing attacks we usually get phishing emails. A Phishing email is a cybercrime that uses deception to steal confidential details about users and organizations. Whenever a phishing email is received, victims are tricked into disclosing information because they trust the source and believe that the party is acting with the best of intentions. They respond immediately to the email without thinking twice. In a phishing email, cybercriminals will typically ask for your: ● Username	**ESR** **Yes!** **ESR** **Yes!** **ESR** **Yes!**

• Date of birth • Social security numbers • Phone numbers • Credit card details • Home address • Password **How Does Phishing Happen?** When a victim replies to an urgent email that demands urgent action, it is considered phishing. Phishing emails may request actions such as: • Clicking an attachment • Macro-enabled Word documents • Password updates • Social media connection request • Using a new wi-fi hot spot	
Open the Teacher Activity 1	*Student opens the Student Activity 1*
You might be interested in learning how attackers usually generate phishing emails Ok before starting just tell me what you noticed in phishing activity? Can you recall that?	**ESR** **Yes!** **ESR**

Yes, we saw the link and the webpage as well, and we always know when something works at frontend, there at the backend it must be code which makes things happen. Let's make a note of our requirements! ● login page for user ● Upon login, store user's information in local storage ● And then we will redirect to the video chat app. Today we will work on our second point frist Let's get started We know all websites or web applications frequently require processing incoming data from users and then post it on the server, a lot of processing like this happens on web applications Data requests can be provided in the form of query strings, forms, and JSON objects. Now our first main task is to access the user data from webpage,, for that we are going to use web framework flask Flask is the python module used to create instances of web application Even Flask, we have used in many previous applications too, As we have seen, all of our applications that we use to run on "localhost:5000" So first of all for creating phishing links , we will build a Flask application that accept either query strings, form data, or JSON objects that later on will return HTML Pages Let's create Flask app first :	**Link, Webpage**

Before that we need to install Flask at our system, if its not installed in system To avoid conflicts with libraries, install Flask in a virtual environment. To create and activate a virtual environment, we can simply run the following commands - **Mac/Ubuntu -** ***python -m venv venv*** ***source venv/bin/activate*** **Windows -** ***python -m venv venv*** ***venv\Scripts\activate.bat*** Next, we will run the following command to install ***flask*** into our virtual environment ***pip install flask***	
The flask framework looks for HTML templates in a folder called ***templates***. So, you should create such an empty folder called "***templates***" and then put all the HTML templates in there. Here is how the web app directory tree should like at this point: Python or .py script stays outside of the templates folder. So first will start working on our main.py file	



In order to access the request object in Flask, we need to import Flask and request

import url_for : ***'url_for***" generates a URL to an endpoint according to the method provided.

import render_template : This function generates output from a template

import jsonify : Handles "JSON" data properly using Flask's ***jsonify()*** method

import csv : reads and writes tabular data in CSV format.

```
from flask import Flask, redirect, url_for, request, render_template, jsonify
import csv
```

We have imported the CSV module to create CSV with the victim's email ID and password that they enter in our Google's Login Form!

Also, to start the application, we would call the main

<table>
<tr><td>function

The debug parameter is set to true. This will help track down possible Python errors on the web page</td><td></td></tr>
<tr><td colspan="2">

```
if __name__ == "__main__":
    app.run(debug = True)
```

</td></tr>
<tr><td colspan="2">Teacher Stops Screen Share</td></tr>
<tr><td colspan="2">STUDENT-LED ACTIVITY - 20 mins</td></tr>
<tr><td colspan="2">

- Ask the student to press the ESC key to come back to the panel.
- Guide the student to start Screen Share.
- The teacher gets into Full Screen.

</td></tr>
<tr><td colspan="2"><u>ACTIVITY</u>

- Student will perform the symmetric Algorithm
- Student will perform the asymmetric Algorithm

</td></tr>
<tr><td>Teacher Action</td><td>Student Action</td></tr>
<tr><td>Guide the student to get the boilerplate code from Student Activity 2</td><td>Student clones the code from Student Activity2</td></tr>
<tr><td>After creating the Flask class, we create a new instance of it. And pass the argument, the “_name_ “.Flask needs this information so it knows where to look for resources such as templates and static files.

With “route()”, we tell Flask which URL should run our</td><td></td></tr>
</table>

function.

Next, we are creating a function "***index"*** that returns the (index.html) that will be created later to design our webpage The function is mapped to the home using **'/' URL.** This means when the user navigates to ***"localhost:5000",*** the home function will run and the output will be displayed on the webpage.

Using the ***"render_template"*** method from the flask framework, we passed an HTML file to the method and it returned to the browser when the user visits the "URL" associated with that template.

```
app = Flask(__name__)

@app.route("/")
def index():
    return render_template("/index.html")
```

In the next step, we will create the login function, whose primary function is to retrieve the username and password and to store the same in the csv format

- Initialize variable ***"username"*** which will request for json data using ***get () method. get()*** *is used to request data from a specified resource.*
- Initialize variable ***"password"*** which will request for json data using ***get ()*** *method*
- By using open() we can access csv files, and by

using "a+" we can append usernames and passwords inside csv files.

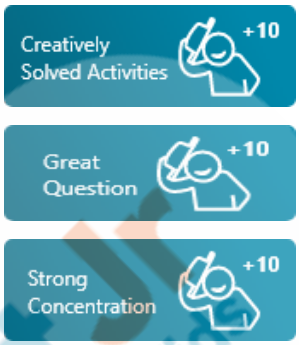
- "CSV" or (Comma-separated values) files are text files that contain a list of values (or fields) separated by commas. CSV is a common data exchange format used by many applications., HTML, JSON and others are also common data exchange formats.
- For writing data inside csv files will use ***"writer()"***
- CSV represents data in tabular form and we want to write data in row format by using **writerow()** method
- ***"writerow()"*** will write username and password inside csv file
- Return json objects
- If data was successfully inserted in csv files, so show status success

```
def login():
    username = request.json.get("username")
    password = request.json.get("password")
    with open("creds.csv", "a+") as f:
        csv_writer = csv.writer(f)
        csv_writer.writerow([username, password])
    return jsonify({
        "status": "success"
    }), 201
```

Now, it's time to run the program

```
 * Serving Flask app 'get' (lazy loading)
 * Environment: production
[31m   WARNING: This is a development server. Do not use it in a production dep
loyment.[0m
[2m   Use a production WSGI server instead.[0m
 * Debug mode: on
 * Restarting with stat
```

Teacher Guides Student to Stop Screen Share	
WRAP UP SESSION - 5 Mins	
Quiz time - Click on in-class quiz	
Question	**Answer**
What is the purpose of the render_template **f**unction? A. Generates output from a template B. Get data from template C. Post data from template D. None of the above	**A**
What is a CSV format? A. Comma segregate value B. Common seperate value C. Comma separated value D. None of the above	**C**
Why do we need the “url_ for” in flask application? A. **url_for** generates a URL to an endpoint B. Act as a url initializer C. Act as a web page D. None of the above	**A**
End the quiz panel	
FEEDBACK • **Appreciate the students for their efforts in the class.** • **Ask the student to make notes for the reflection journal along with the code they wrote in today's class.**	
Teacher Action	**Student Action**

You get Hats off for your excellent work! In the next class	*Make sure you have given at least 2 Hats Off during the class for:* 
Project Discussion	
Teacher Clicks 	
ADDITIONAL ACTIVITIES	
Additional Activities *Encourage the student to write reflection notes in their reflection journal using markdown.* Use these as guiding questions: ● What happened today? ○ Describe what happened. ○ The code I wrote. ● How did I feel after the class? ● What have I learned about programming and developing games? ● What aspects of the class helped me? What did I find difficult?	*The student uses the markdown editor to write her/his reflections in the reflection journal.*

ACTIVITY LINKS		
Activity Name	**Description**	**Link**
Teacher Activity1	Phishing Link	https://en.wikipedia.org/wiki/Phishing
Teacher Activity 2	Reference Code	https://github.com/pro-whitehatjr/Pro_C224_Teacher-reference-code
Student Activity 1	Phishing Link	https://en.wikipedia.org/wiki/Phishing
Student Activity 2	Boilerplate Code	https://github.com/pro-whitehatjr/PRO-C224_StudentBoilerCode