Welcome to the Monogram

Shakeshka Alzhappar, Susan Wang, Royce Gopaul
st4454, sw5571, rg4668

Motivations

Improving User Experience

- Security
- Aesthetics
- Accessibility
- Entertainment

Demo: App Functions

Security

- Encrypted messages
- Login with unique password for every user

Fun Features

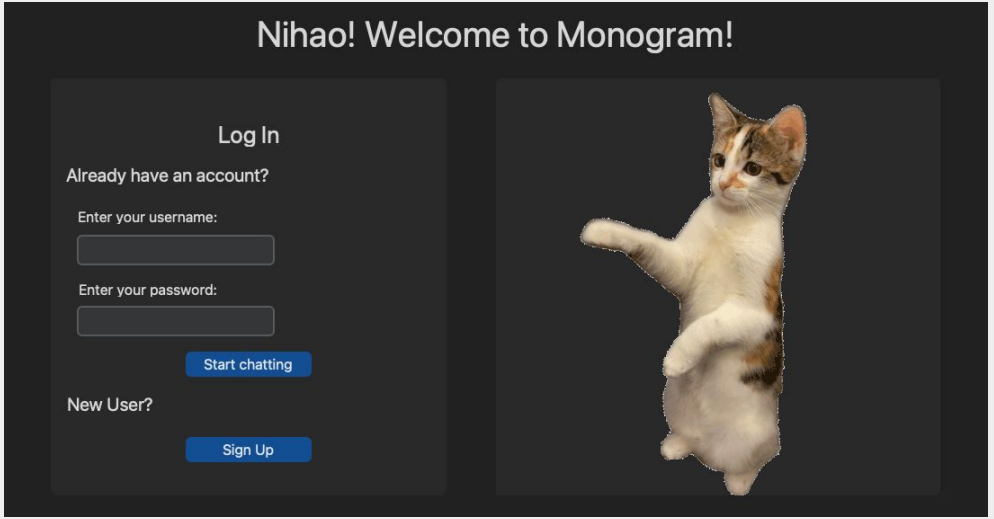
- Light/Dark Modes
- Music On/Off
- Snake Game
- Scoreboard

Discussion:

GUI/Chat System

GUI/Chat System: Login/Signup

```
def usrLogin(self):
    global usrName
    usrName = self.input_username.get()
    usrPwd = self.input_password.get()
    try:
        with open('usrs_info.pickle', 'rb') as usrFile:
            usrsInfo = pickle.load(usrFile)
    except FileNotFoundError: # if file doesnt exist, create a new file
        with open('usrs_info.pickle', 'wb') as usrFile:
            usrsInfo = {'admin': 'admin'}
            pickle.dump(usrsInfo, usrFile)
    if usrName in usrsInfo:
        if usrPwd == usrsInfo[usrName]:
            messagebox.showinfo('Successfully Login!', 'Welcome to the chat system ' + usrNa
            client.login(usrName)
            self.startChatting()
def usrSignUp(self):
    def signUp():
        password = newPassword.get()
        passwordConfirm = newPasswordConfirm.get()
        name = signName.get()
        with open('usrs_info.pickle', 'rb') as usrFile:
            userlist = pickle.load(usrFile)
        if password != passwordConfirm:
            messagebox.showerror('Passwords do not match! Please try again!')
        elif name in userlist:
            messagebox.showerror('You have signed up! Please login with your username.')
        elif name == "":
            messagebox.showerror("Forget to enter your name?")
        elif password == "":
            messagebox.showerror("Forget to enter your password?")
        else:
```



GUI/Chat System: GUI Settings

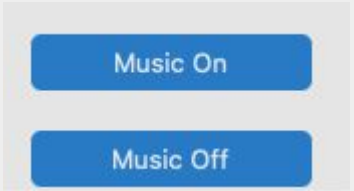
```
def change_appearance_mode_event(self, new_appearance_mode: str):
    set_appearance_mode(new_appearance_mode)

def change_scaling_event(self, new_scaling: str):
    new_scaling_float = int(new_scaling.replace("%", "")) / 100
    set_widget_scaling(new_scaling_float)
```

GUI/Chat System: Music!!!

```
def second_column_music_on(self):
    mixer.init()
    mixer.music.load("music.mp3")
    mixer.music.play()

def second_column_music_off(self):
    mixer.music.pause()
```

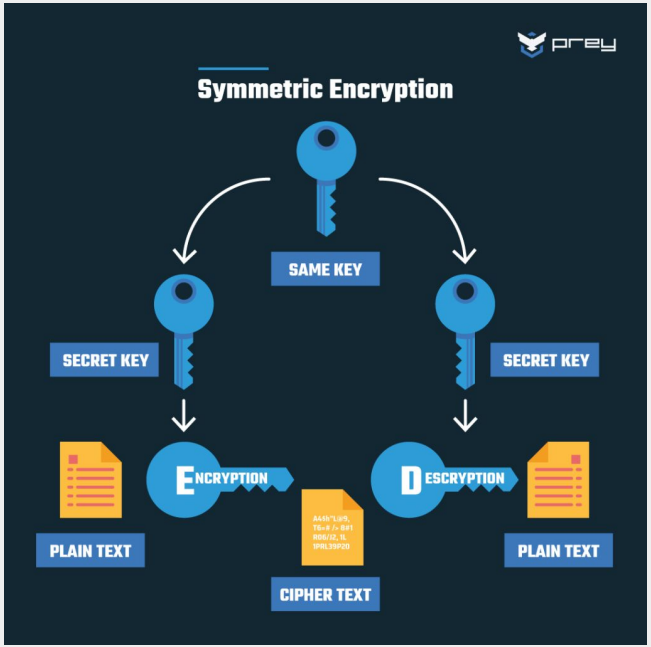


Discussion:

Secure Messaging

Secure Messaging: Encryption

```python
def encryption(message,public_key):
    e,n = int(public_key[0]), int(public_key[1])
    encrypted_number = []
    #message += "xxx"
    for i in message:
        encrypted_number.append(quick_pow(ord(i),e,n))
    print(str(encrypted_number))
    return str(encrypted_number)
```



Secure Messaging: Encryption

```python
def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a


def quick_pow(x,y,n):
    if y == 0:
        return 1
    if y == 1:
        return int(x)**y%n
    if y%2 == 0:
        return (quick_pow(int(x),y//2,n)**2)%n
    return (x * (quick_pow(int(x),y//2,n)**2)%n)%n
```

```python
def find_multiplicative_inverse(a,b):
if gcd(a,b) != 1:
    return None
i,j,k = 1,0,a
x,y,z = 0,1,b
while z != 0:
    q = k//z
    x,y,z,i,j,k = (i-q*x), (j-q*y),(k-q*z),x,y,z
return i % b
```

```python
def is_prime(n):
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True

def generate_prime_list(a,b):
    prime_list = []
    for i in range(a,b):
        if is_prime(i):
            prime_list.append(i)
    return prime_list
```

```python
def decryption(message,private_key):
    d,n = int(private_key[0]), int(private_key[1])
    decrypted_message = []
    if len(message) > 0:
        message = str(message)
    message = message[1:-1]
    message = message.split(",")

    for i in message:
        t = int(i)
        decrypted_message.append(chr(quick_pow(t,d,n)))
    message_text = "".join(decrypted_message)
    return message_text
```

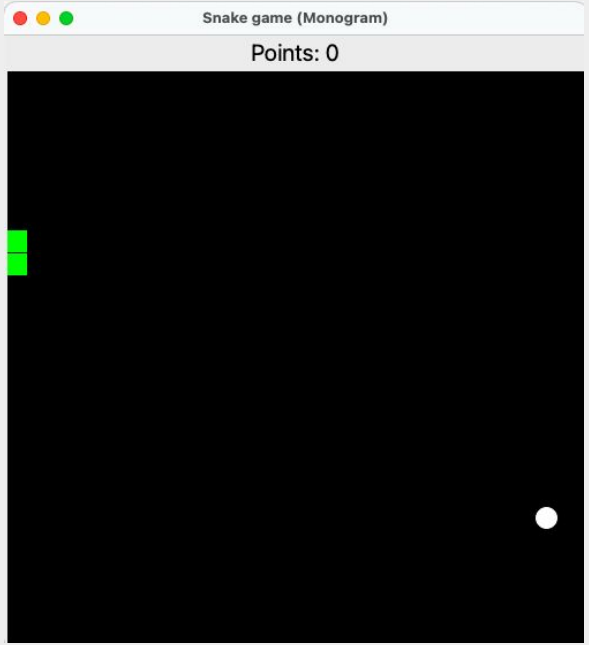
```python
def generate_key():
    p = random.choice(prime_list)
    q = random.choice(prime_list)
    while p == q:
        q = random.choice(prime_list)
    n = p*q
    # calculate euler function phi(n)
    phi = (p-1) * (q-1)
    # Choose an integer e that is relative prime
    e = random.randint(1, phi)
    while gcd(e, phi) != 1:
        e = random.randrange(1, phi)
    # Use Extended Euclid's Algorithm to generat
    d = find_multiplicative_inverse(e, phi) #d =
    public_key = (e,n)
    private_key = (d,n)
    return public_key, private_key
```

Discussion:

Online Gaming

Online Gaming: Snake Game

```
def start_game(self):
    runpy.run_path(path_name='snake.py')
    print("Attempting to read score file...")
    try:
        with open('score.txt', 'r') as f:
            global score
            score = f.read().strip()
        print(f"Score: {score}")
        self.load_score()
    except:
        print("No score found.")
```



Online Gaming: Scoreboard


```python
def scoreboard(self):
    # pop up the sign up window on the top
    self.scoreboard_window = Toplevel(self)
    self.scoreboard_window.title('Scoreboard')
    self.scoreboard_window.geometry(f"{400}x{600}")
    self.scoreboard_window.resizable(False, False)

    # Column labels
    CTkLabel(self.scoreboard_window, text='#', width=120, height=25).grid(row=0, column=0)
    CTkLabel(self.scoreboard_window, text='Username', width=120, height=25).grid(row=0, column=1)
    CTkLabel(self.scoreboard_window, text='Highest Score', width=120, height=25).grid(row=0, column=2)

    # Load the data from the .pickle file
    with open('scoreboard.pickle', 'rb') as file:
        global scores
        scores = pickle.load(file)

    # Function to determine sort key
    def sort_key(item):
        username, score = item
        try:
            # Attempt to convert score to integer
            return (True, int(score))  # True means it's a valid integer, sort by negative value for descending order
        except ValueError:
            # Return False to put non-integers last, keep original order among non-integers
            return (False, 0)

    # Sort the scores dictionary by attempting to convert scores to integers, with alphabetic last
    sorted_scores = sorted(scores.items(), key=sort_key, reverse=True)

    # Display scores in the grid, adding a rank number
    for i, (username, score) in enumerate(sorted_scores, start=1):
        CTkLabel(self.scoreboard_window, text=str(i), width=120, height=25).grid(row=i, column=0)
        CTkLabel(self.scoreboard_window, text=username, width=120, height=25).grid(row=i, column=1)
        CTkLabel(self.scoreboard_window, text=str(score), width=120, height=25).grid(row=i, column=2)
```

```python
def load_score(self):
    # loading existing scores or initialize an empty dictionary
    print("Loading the score into .pickle")
    try:
        with open('scoreboard.pickle', 'rb') as pf:
            user_scores = pickle.load(pf)
    except (FileNotFoundError, EOFError):
        user_scores = {}  # creating if no file exists or file is empty

    # updating the score if the user exists and the new score is higher, or add new user
    if usrName in user_scores:
        if int(score) > int(user_scores[usrName]):
            user_scores[usrName] = score
            print(f"Updated score for {usrName} to {score}")
    else:
        user_scores[usrName] = score
        print(f"Added new user {usrName} with score {score}")

    # saving the updated dictionary back to the pickle file
    with open('scoreboard.pickle', 'wb') as pf:
        pickle.dump(user_scores, pf)
```

Analysis:

Organization & Defined Classes

Analysis:

Room for Improvement

Thank you!!!