Login Page	register_get()	**Intention:** This method sends a request to the server to open the register page (register.html) when the "Register" button is clicked
		Inputs: Left click on the "Register" button on the login page to activate the GET method and request the register.html page
		Outputs: The regiter.html page opens
	login_post()	**Intention:** Sends login information inputted by user to the server.
		Inputs: User login information (email and password)
		Outputs: If the login information is contained in the back end, then a GET request is sent to the server to open the home page (so the homepage opens as a result). If the inputted login information does not exist in the back end, then the login page is kept open with an error message, "login failed".
Register Page	register_post()	**Intention:** Send registration inputs to the server and confirm those inputs meet the requirements of R2.
		Inputs: email, name, password, and password2
		Output: If the registration inputs meet the requirements of R2, then a GET request is sent to the login page and the login page is opened as a result. If the registration inputs do not meet the requirements of R2, then an error message appears on the registration page informing the user of their error (the passwords do not match, email format error, etc.)
	login_get()	**Intention:** Sent GET request to login.html when the "Login" button on the registration page is clicked
		Inputs: left click on the "Login" button
		Output: The login.html page opens
Home Page	logout()	**Intention:** logout of existing session
		Inputs: Left click on "Logout" button on the homepage
		Output: Sends a GET request to the server to open the login page
	authenticate(inner_function)	**Intention:** Once the user's logged in, maintain that the user is logged in so when the user submits a form or moves from one page to another, the system does not have to recheck that they are logged in
		Inputs: an inner function that accepts a user object
		Outputs: none
	profile(user)	**Intention:** Show the profile page for that user with all the available tickets and their information (owner's email address, quantity, price, etc.) listed
		Inputs: User
		Outputs: show index.html for that user
Other	page_not_found(error)	**Intention:** For any other URL request that are not identified in R1 through R7, show a 404-error page

		Inputs: error
		Outputs: Open the 404-error page
	check_empty_fields(field)	**Intention:** check if login input fields are empty
		Inputs: field (the email and password inputs)
		Outputs: If a field is left empty, output error message on login page, “Field is required”
	is_valid_email(email)	**Intention:** Check if the email address meets the requirements described in R1 and R2
		Inputs: email
		Outputs: True if email is valid, False if email is invalid
	Is_valid_password(password)	**Intention:** Check if password meets requirements of R1 and R2
		Inputs: password
		Outputs: True if password is valid, False if password is not valid
	is_valid_user(name)	**Intention:** Confirm that the name inputted meets the requirements of R1 and R2
		Inputs: name
		Outputs: True if name is valid, False otherwise