

A Murder Mystery Website

Amelia Handley [40326169]

¹Edinburgh Napier University
40326169@napier.ac.uk

1 Introduction

The aim of this coursework was to create and implement the website that has been planned using python and the python flask framework. Additional libraries which have been used to build the website have been detailed. Features which were not included and features which, if there were an opportunity, have also been described.

2 Description of final site

The result was a functional website based on the initial plans made for the murder mystery site. The website features a home page which gives a description of the game, a page to allow the user to create an account and a page to log the user into the website after they have signed-up (Appendix 1). Once they have be logged in the user will be able to access and play the main game. The game uses a simple user interface to make it easy to navigate and understand.

3 Differences from initial plan

Not all the features and libraries outlined in the initial report were used or were changed on implementation.

3.1 Databases

Initially, the database which was going to be used to store the user information was going to be used was SQLite3. However, MySQL was used in the final website. This is because the initial use for the database was for storing the information to allow a user to log into the website. MySQL has a lot of useful security features that are inbuilt such as authentication for user login and for passwords [1].

3.2 User Profile and Forum Pages

Outlined in the initial plan was a feature to allow a user to update and access their own private information as well as a forum page to discuss gameplay. However, these features were not required with the website produced as the user only needs access to the game - although it would still be a nice additional feature.

4 Additional Libraries used

4.1 flask_mysqldb

To allow the python flask to connect to the MySQL database created required the additional flask plug Flask-MySQLdb was installed and imported in the main.py[1]. To understand how the database could connect to the main application a Youtube series was used [2].

4.2 wtForms

WTForms are a framework which can help add the creation of a python (flask) website. use WTForms you define what your form as a classes first, so in the case of this course-work a class was created for the registration of a user [3].

```
class RegisterForm(Form):
    username = StringField('Username', [validators.Length(min=1, max=20)])
    password = PasswordField('Password', [validators.Length(min=3, max=20)])
```

Image 1. Registration Form from app.py

```
@app.route('/create', methods=['GET', 'POST'])
def create():
    form = RegisterForm(request.form)
    if request.method == 'POST' and form.validate():
        username = form.username.data
        password= sha256_crypt.encrypt(str(form.password.data))

        # Create Cursor
        cur = mysql.connection.cursor()

        cur.execute("INSERT INTO users(username, password) VALUES(%s, %s)", (username, password))

        # Commit to DB
        mysql.connection.commit()

        # Close connection
        cur.close()

        flash('You have now created an account', 'success')

        return redirect(url_for('login'))
    return render_template('create.html', form=form)
```

Image 2: Create a user function from app.py

From Image 1 we describe a user as needing a username and password. Using the “StringField” method we save the username and using the “PasswordField” we save the password. Validators, imported as validators, are then used to ensure that the data has been entered within the range specified [3].

Then to create the form, if the data has been submitted using a HTTP ‘POST’ method. The data is then validated using the validate() method which ensures that the data meets the validators requirements.

```
{% macro render_field(field) %}
    {{ field.label }}
    {{ field(**kwargs)|safe }}
    {% if field.errors %}
        {% for error in field.errors %}
            <span class="help-inline">{{ error }}</span>
        {% endfor %}
    {% endif %}
{% endmacro %}
```

Image 3: Form helper from ‘_formHelpers.html’

To pass the form to be read by the templates (Image 3). It accepts keyword arguments which are forwarded to the WTForm’s field functions. This will then render the field for us [3].

```
{% extends 'layout.html' %}

{% block body %}
<div class="box">
    <h1>Create An Account</h1>
    {% from "includes/_formHelpers.html" import render_field %}
    <form method="POST" action="">
        <div class="form-group">
            {{render_field(form.username, class_="form-control")}}
        </div>
        <div class="form-group">
            {{render_field(form.password, class_="form-control")}}
        </div>
        <p><input type="submit" class="btn btn-primary" value="Submit"></p>
    </form>
{% endblock %}
```

Image 4. Creating a user from ‘create.html’

Using the form helper, the render_field is created. So, this means that a class will be added to the input element, for instance the username or the password (Image 4) [3].

4.3 passlib.hash

Passlib.hash is a password hashing framework which can be used to securely store a users personal information (Image 5). Specifically, the sha256_crypt was imported into the program and used to hash the password.

```
mysql> SELECT * FROM users;
+-----+-----+-----+-----+
| id | username | password | register_date |
+-----+-----+-----+-----+
| 27 | Amelia | $5$rounds=535000$Q1KiH5z55HkGQnwe$b4jkaniVGA.OQDeq.89G.uZf/I9EC6wumGW24O..zY/ | 2019-12-04 13:38:01 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Image 5. Entry into database, from users database (myflaskapp).

An additional benefit of using passlib.hash is that it includes functions such as .verify which were used to validate the user when they were logging into the website.

4.4 Flask flash

A goal of the website was to create a simple but efficient user experience. Flask flash is an additional library that can be used to provide feedback to the user using messages. These messages have been encoded to appear at the top of the file [5] .

```
{% with messages = get_flashed_messages(with_categories=true)%}
    {% if messages %}
        {% for category, message in messages %}
            <div class="alert alert-{{ category }}">{{ message }}</div>
        {% endfor %}
    {% endif %}
{% endwith %}

{% if error %}
    <div class="alert alert-error">{{ error }}</div>
{% endif %}

{% if warning %}
    <div class="alert alert-warning">{{ warning }}</div>
{% endif %}

{% if success %}
    <div class="alert alert-success">{{ message }}</div>
{% endif %}
```

Image 6. Flash messages from ‘_flashMessages.html’

These messages will help prompt the user when using the website (Image 6). So it will tell the user with messages such as “You have created an account” or “No user found” [2].

5 Extra Features

Outlined in the initial document was for the website to enable to interact with others. However, they were unable to be completed in time for submission. Therefore, additional features the website which would be a justified extension of the current website would be a forum page. This would enable to discuss their own gameplay experiences. This could also be implemented as a comments section at the end of the game.

Another feature would be to allow users to create their own games. This would mean creating another database to store and update the created games.

6 Challenges

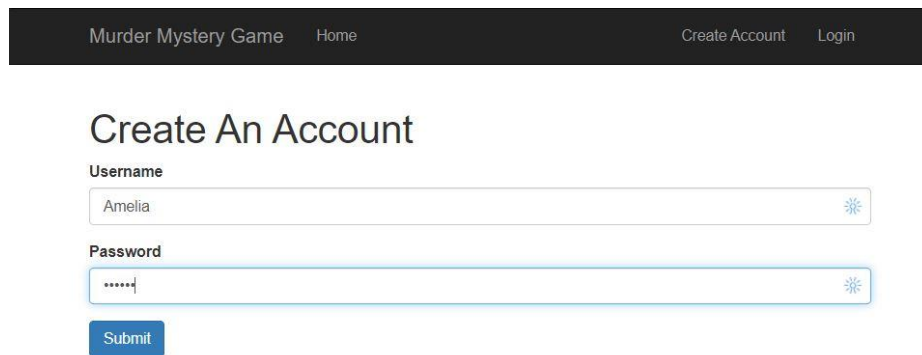
One of the main challenges when creating this website was to figure out a method to log in a user. Additionally, creating a database to store the game was difficult so instead the game uses the routes created in the main application to navigate.

7 Achievements

The game website that was produced was satisfactory using the python flask framework. This project helped achieve a further insight into website development in a fun way.

Appendices

Appendix 1



Murder Mystery Game Home Create Account Login

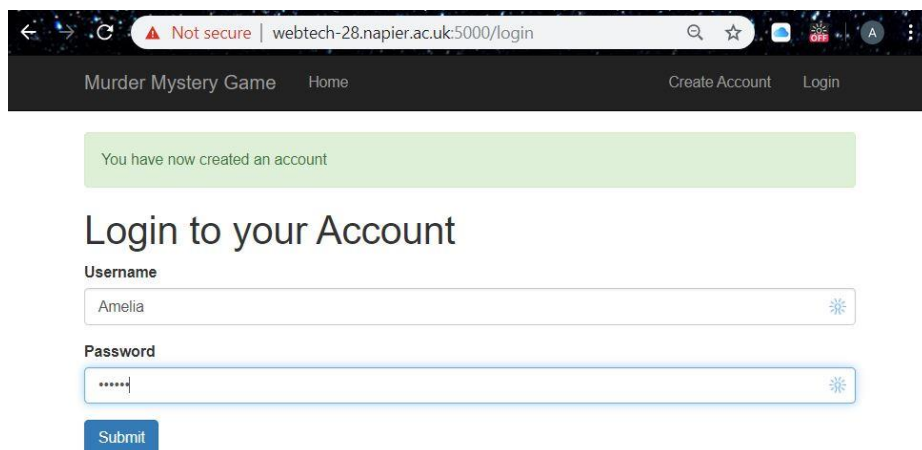
Create An Account

Username

Password

Submit

Image 7. Website Registration Page



Not secure | webtech-28.napier.ac.uk:5000/login

Murder Mystery Game Home Create Account Login

You have now created an account

Login to your Account

Username

Password

Submit

Image 8. Log in Page

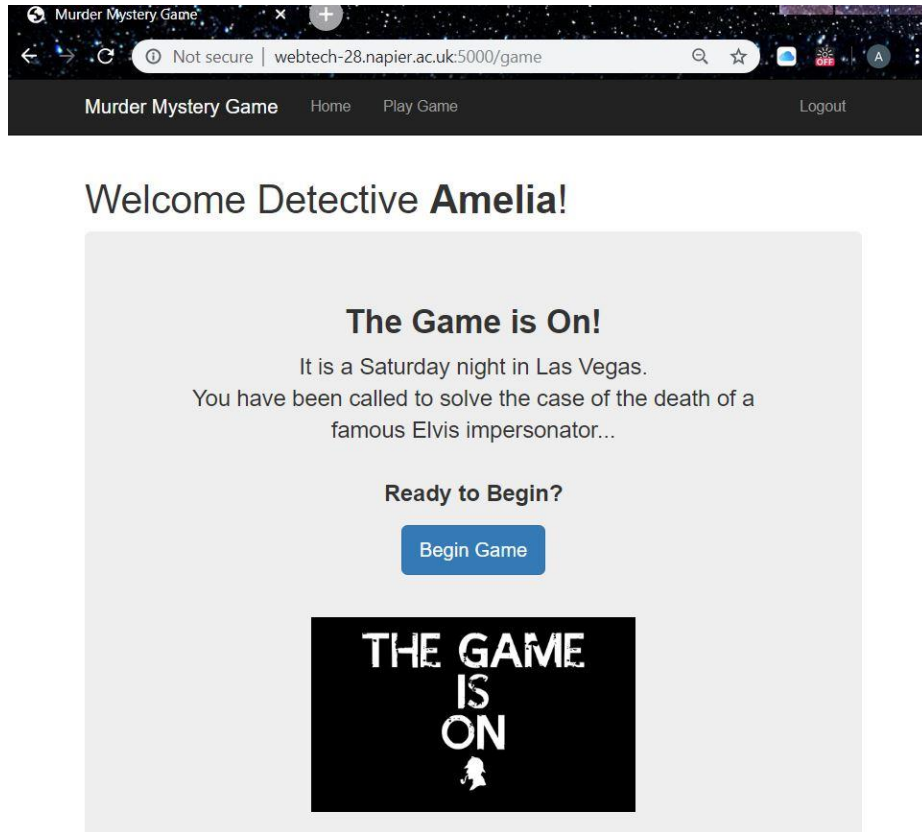


Image 9. Game Page

References

1. Flask-MySQLdb's documentation, <https://flask-mysqldb.readthedocs.io/en/latest/>, last accessed 2019/12/01
 2. Traversy Media., <https://www.youtube.com/watch?v=addnlzdSQs4&t=27s>, last accessed 2019/12/01
 3. Form Validation with WTForms, <https://flask.palletsprojects.com/en/1.1.x/patterns/wtforms/>, last accessed 2019/12/01
 4. Passlib, https://passlib.readthedocs.io/en/stable/lib/passlib.hash.sha256_crypt.html, last accessed 2019/12/01
 5. Message Flashing, <https://flask.palletsprojects.com/en/1.1.x/patterns/flashing/>, last accessed 2019/12/01
- Flask Session, <https://pythonhosted.org/Flask-Session/>, last accessed 2019/12/01