# Tuesday: User Profile

## Creating a Profile page

We can now create a profile for our new user. But first we need to update `User` so we can store more properties.

*app/models.py*

```
class User(UserMixin,db.Model):
    __tablename__ = 'users'

    id = db.Column(db.Integer,primary_key = True)
    username = db.Column(db.String(255),index = True)
    email = db.Column(db.String(255),unique = True,index = True)
    role_id = db.Column(db.Integer,db.ForeignKey('roles.id'))
    bio = db.Column(db.String(255))
    profile_pic_path = db.Column(db.String())
    password_secure = db.Column(db.String(255))

    #....
```

We add two new columns; one for the `bio` which is the users biography and the other for `profile_pic_path` to store the path of the profile photo. We can update our database by creating an updated migration file and upgrade the database.

Now let's create a profile view function.

*app/main/views.py*

```
from flask import render_template,request,redirect,url_for,abort
from ..models import Reviews, User

#.....
@main.route('/user/<uname>')
def profile(uname):
    user = User.query.filter_by(username = uname).first()

    if user is None:
        abort(404)

    return render_template("profile/profile.html", user = user)

#.....
```

We first import the `abort` function, that stops a request, and returns a response according to the status code passed in. We then create a dynamic route that is handled by the `profile` view function.

We then query the database to find the `user` according to the username passed. If no user is found the `abort` is called and a `404` status code is returned as a response. If a `user` is found we render a template and pass in the `user` as a variable.

*app/templates/profile/profile.html*

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

{% block content %}
<div class="container">
<div class="row">
<!-- displaying  Username-->
  <div class="col-md-4">
    <h3> {{user.username| capitalize}} </h3>

    <!--Displaying user bio-->
    {% if user.bio %}
      <p> {{user.bio}}</p>
    {%else%}
      <p> {{user.username| capitalize}} has no bio </p>
    {% endif %}
  </div>

  <!--Display profile photo-->
  <div class="col-md-4">

    {% if user.profile_pic_path %}
    <img src="{{url_for('static',filename=user.profile_pic_path)}}" alt="" class="img-thumbna
il">
    {%else%}
    <p>No profile picture</p>
    {% endif %}
  </div>
</div>
</div>
{% endblock %}
```

We create a template file for our profile where we display our username, bio and profile picture of the User.

*app/templates/navbar.html*

```
....
                <ul class="nav navbar-nav navbar-right">
                  {% if current_user.is_authenticated %}
                  <li><a href="{{url_for('main.profile',uname=current_user.username)}}">Profi
le</a></li>
                  <li><a href="{{url_for('auth.logout')}}">Sign out</a></li>
                  {% else %}
                  <li><a href="{{url_for('auth.login')}}">Sign in</a></li>
                  {%endif%}
                </ul>

....
```

We then create a link on our navigation bar to direct us to the profile page. This link will be visible only if a user is authenticated.

# Edit Profile

We want to update our bio to say something interesting about us.

*app/main/forms.py*

```
#....
class UpdateProfile(FlaskForm):
    bio = TextAreaField('Tell us about you.',validators = [Required()])
    submit = SubmitField('Submit')
```

We create a new form class `UpdateProfile` that has only the `bio` Textarea form field.

*app/main/views.py*

```
#....
from .forms import ReviewForm,UpdateProfile
from .. import db

#....
```

We import the new form class into our _views.py_ module in our main blueprint. Also, we import db from our app since we will need it when saving profile information changes to the database.

We can now create a view function that will handle an update request

*app/main/views.py*

```
@main.route('/user/<uname>/update',methods = ['GET','POST'])
@login_required
def update_profile(uname):
    user = User.query.filter_by(username = uname).first()
    if user is None:
        abort(404)

    form = UpdateProfile()

    if form.validate_on_submit():
        user.bio = form.bio.data

        db.session.add(user)
        db.session.commit()

        return redirect(url_for('.profile',uname=user.username))

    return render_template('profile/update.html',form =form)
```

We create a `update_profile` view function that takes in a username, and instantiates the `UpdateProfile` form class.

We query the database to find a `user` with the same `username`.

If the form is validated we update the content of the `user.bio` property to fill in what the user has submitted and redirect the user back to the profile page where he can see the new bio.

If not we render the _update.html_ template and pass in the form instance.

*app/templates/profile/profile.html*

```
{% if user == current_user %}
    <a href="{{url_for('main.update_profile',uname=user.username)}}">Edit profile</a>
{% endif %}
```

We create a link to the `update_profile` view function inside our _profile.html_ file.

*app/templates/profile/update.html*

```
{% extends 'base.html' %}
{% import 'bootstrap/wtf.html' as wtf %}

{% block content %}
<div class="container">
```

```
  {{wtf.quick_form(form)}}
</div>
{% endblock %}
```

Inside our *update.html* template we call the `wtf.quick_form()` function and pass in the form instance. This will display our update bio form with bootstrap styling.

You can find the application at this point from here https://github.com/mbuthiya/watchlist/tree/26-Creating-a-user-profile