

Flask 2: User detail

User detail

- To view user detail, we will use a route parameter.
- We can write a separate function 'view_user' that uses the same method and route.
- It is technically the same endpoint, but now it has an optional parameter

```
@app.get('/users/<username>')  
def view_user(username):  
    return
```

Templates

- Update the users.html template so that each item or row is a link to a user detail page.
- Don't forget to use url_for
- It should look something like this:

```
<a href="{ { url_for('view_user',username=user.username) } }"> link text </a>
```

Template

- Now create the template for viewing one user (user.html)

User detail (ctd)

- Complete the `view_user` function so that it will:
 - get user data for one user
 - pass the data to `render_template` and return.
- Note:
 - you can use mock data here
 - but it won't help with your routing, only building your template

Getting user by username

- Recall, we wrote an endpoint GET user/{userId}
- What if we want to get by username?

The hack we used (in user API)

- In read users, we accessed the request object directly to get all the query parameters.
- This works, but the option won't show up in your spec

```
@app.get('/users')
async def read_users(req:Request) -> UserCollection:
    '''read users and return result'''

    query = dict(req.query_params)
    return um.read_users(query)
```

Specifying query parameters

- If we specify the query parameters, they will show up in the spec

```
@app.get('/users')
async def read_users(username=None) -> UserCollection:
    '''read users and return result'''

    query = {'username':username}
    return um.read_users(query)
```


Getting users by username

- the GET users endpoint returns a UserCollection
- we need to get our found user

```
res = rq.get(url + '/users', params={
    'username': "joe"})
# self.assertEqual(res.status_code, 200)
uc = json.loads(res.text)
us = uc.get('users')
# self.assertTrue(len(us)==1)
u = us[0]
```