

A decorative header featuring four overlapping spheres: a green one on the left, and blue, red, and yellow ones on the right. A thin black horizontal line is positioned below the spheres.

Part 2: Users API + Images API



Recap: Part 1

- We created a basic wiki
- We revised the wiki to include revisions by redefining our data model to use ReferenceProperties

Adding User Information

- Google App Engine provides a Users API, which allows you to authenticate users with their Google Account (or your Google Apps for your Domain user's accounts)
- This allows you to have access to a user's email and nickname
- You can generate login and logout urls with a simple API call
- You can restrict pages to require login



Generating Login/Logout Urls

An API call allows you to easily generate URLs for a user to sign in and out of your web application.

```
user = users.get_current_user()

if user:
    users.create_logout_url(self.request.path)
else:
    users.create_login_url(self.request.path)
```



Restricting pages to require login

In your `app.yaml` you can specify that a login is required, or restrict it further to only admins:

```
handlers:  
- url: /  
  script: home.py  
  login: required  
  
- url: /myadmin  
  script: admin.py  
  login: admin
```



Images API

- The Image manipulation API provides transforms which allows you to manipulate and store images
 - `resize()`
 - `crop()`
 - `rotate()`
 - `horizontal_flip()`, `vertical_flip()`
 - `im_feeling_lucky()`

Image example

```
photo = self.request.get("my_photo")
```

```
img = images.Image(photo.full_size_image)
```

```
img.resize(width=80, height=100)
```

```
img.im_feeling_lucky()
```

```
thumbnail = db.Blob(img.execute_transforms  
( output_encoding=images.JPEG))
```



Adding User Pages to our Wiki

For our wiki example we created a user model to store information about our wiki authors:

```
class WikiUser(db.Model):  
    wiki_user = db.UserProperty()  
    joined = db.DateTimeProperty(auto_now_add=True)  
    about = db.TextProperty()  
    wiki_user_picture = db.BlobProperty()  
    user_feed = db.LinkProperty()
```



Hack Session 2

Add the handlers:

```
(' /user/ ([^/]+) ', UserProfileHandler)  
( ' /edituser/ ([^/]+) ', EditUserProfileHandler),  
( ' /getphoto/ ([^/]+) ', GetUserPhotoHandler)
```



Hack Session 2 Hints

Use the images API to resize the user submitted photo

Set up a separate handler to retrieve the photo, and serve it with the correct content type

For more information, read the article:

Serving Dynamic Images with Google App Engine

At:

<http://code.google.com/appengine/articles>

