web2py

ideas we borrowed, ideas we had

Massimo Di Pierro

Associate Professor

School of Computing

DePaul University

DEPAUL UNIVERSITY

Contributors

Resources

videos

60+ example apps

web2py.com

github
SOCIAL CODING

web2py.com/appliances

plu

es

dev.s-cubism.com/web2py_plugins

http://www.web2pyslices.com
(built with web2py-based social network software: http://movu.ca/)

Ideas we borrowed

- Model View Controller on WSGI (like everybody else)
- w2p files (like Java's Web application ARchives)
- Optional Routing (like Rails)
- Routing mechanism (like Django)
- Pure Python Template Language (like Mako)
- Helpers (like Rails) but easier to remember: SPAN, A, ...
- request['xxx'] == request.xxx (like web.py)
- web based database interface (like Django admin)

Ideas we had ...

- A tool for both technical and non-techincal people
- Always backward compatible (since 2007, 2.5, 2.6, 2.7, pypy, jython)
- One click deploy (Windows and Mac binaries, USB drive)
- No configuration, no dependencies, and secure by default
- Everything has default (DRY)
- Multi project / multi db / share nothing by default
- Web based IDE (development, editor, deployment, management, translations, testing, debugger, version control) shell optional
- Automatic DB migrations (CREATE and ALTER table)
- Plugins / Components / Ajax with Digitally Signed URLs

... Ideas we had

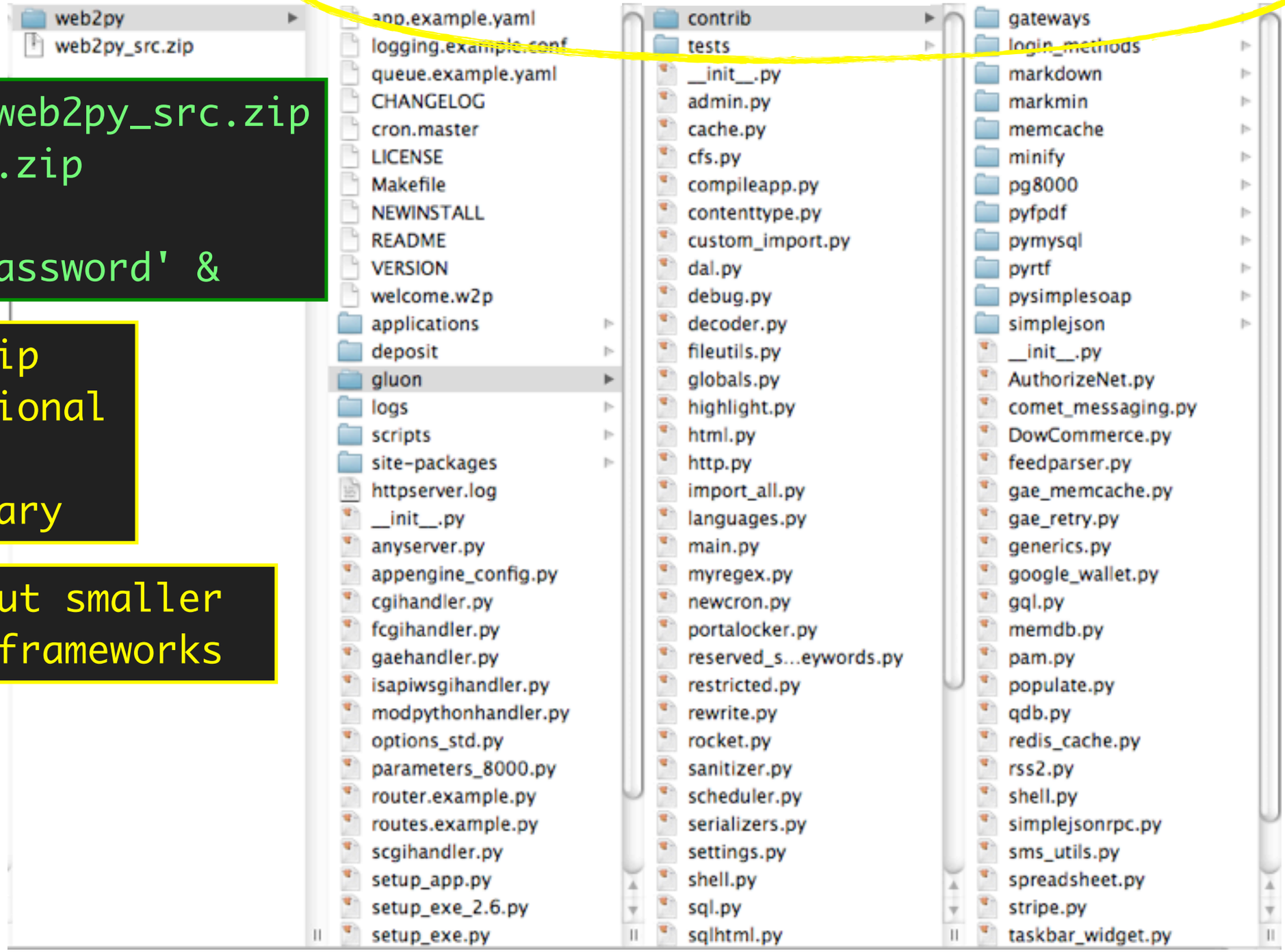
- Role Based Access Control with pluggable authentication modules (openid, dlap, cas, oauth 1/2, pam, janrain, google, dropbox)
- Every app is a Central Authentication Service provider (and optionally consumer)
- Built-in portable cron and master/workers task scheduler
- multi tenant apps with common fields & common filters
- record versioning (without breaking refs)
- embeddable crud & grid controls
- Package lots of 3rd party modules: pyfpdf, database drivers, credit card payment APIs, ...

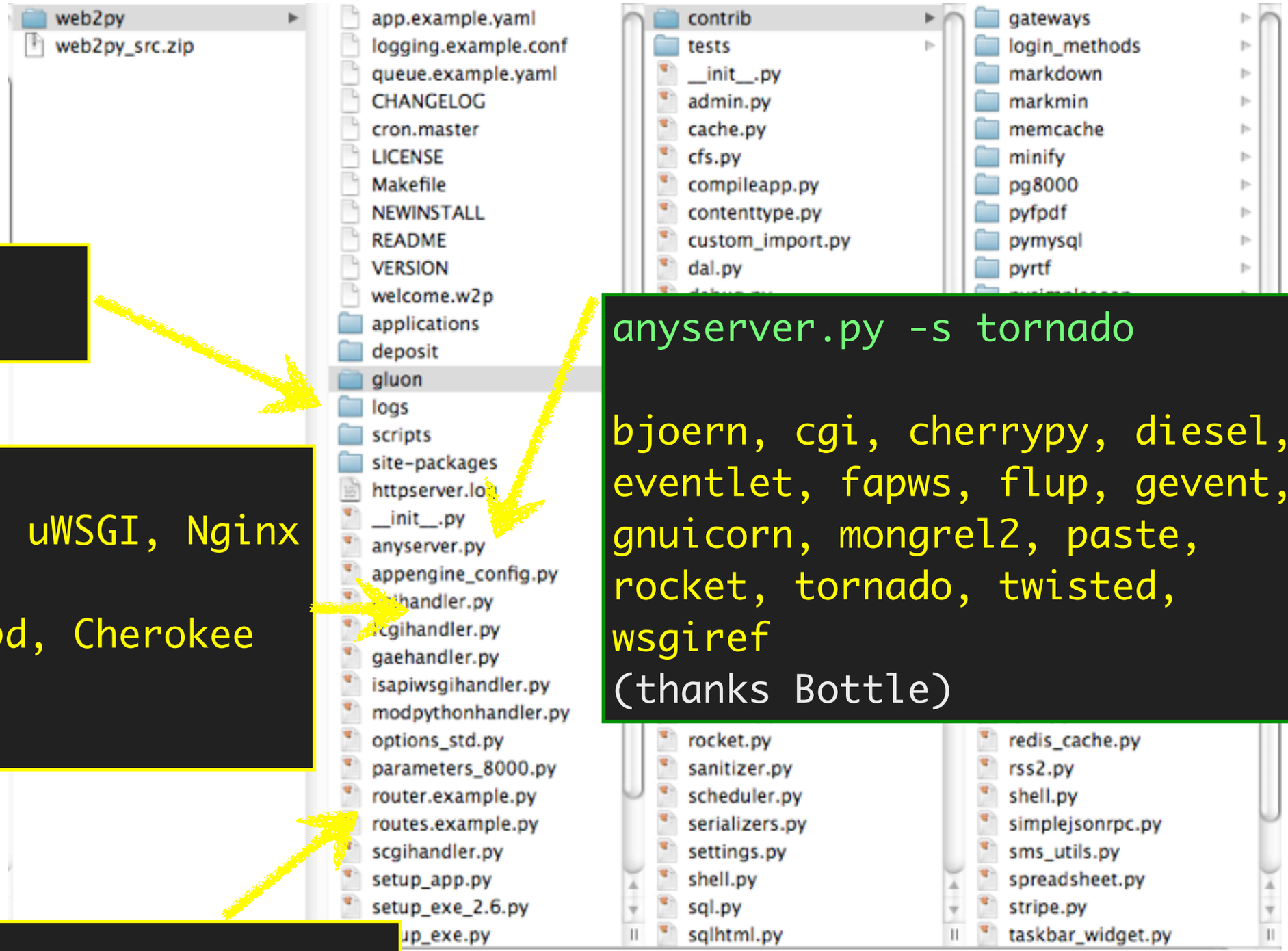
```
wget http://.../web2py_src.zip
unzip web2py_src.zip
cd web2py
web2py.py -a 'apassword' &
```

download and unzip
installation optional
no configuration
no coding necessary

all-inclusive, but smaller
than most micro-frameworks

web2py/ 0.1MB gluon/ 1.3MB contrib/ 2.0MB





*.log:
log files

adapters:
- WSGI: Apache, uWSGI, Nginx
- CGI
- FCGI: Lighttpd, Cherokee
- GAE
- ISAPI: IIS

anyserver.py -s tornado

bjoern, cgi, cherrypy, diesel, eventlet, fapws, flup, gevent, gnuicorn, mongrel2, paste, rocket, tornado, twisted, wsgiref
(thanks Bottle)

routes.py:
like urls.py but optional

dal.py: Database Abstraction Layer
single file / no dependencies / framework agnostic

supports: sqlite, postgresql, mysql, db2, firebird, mssql, google datastore, interbase, ingres, imap, ~sapdb, ~cubrid, ~teradata, ~mongodb

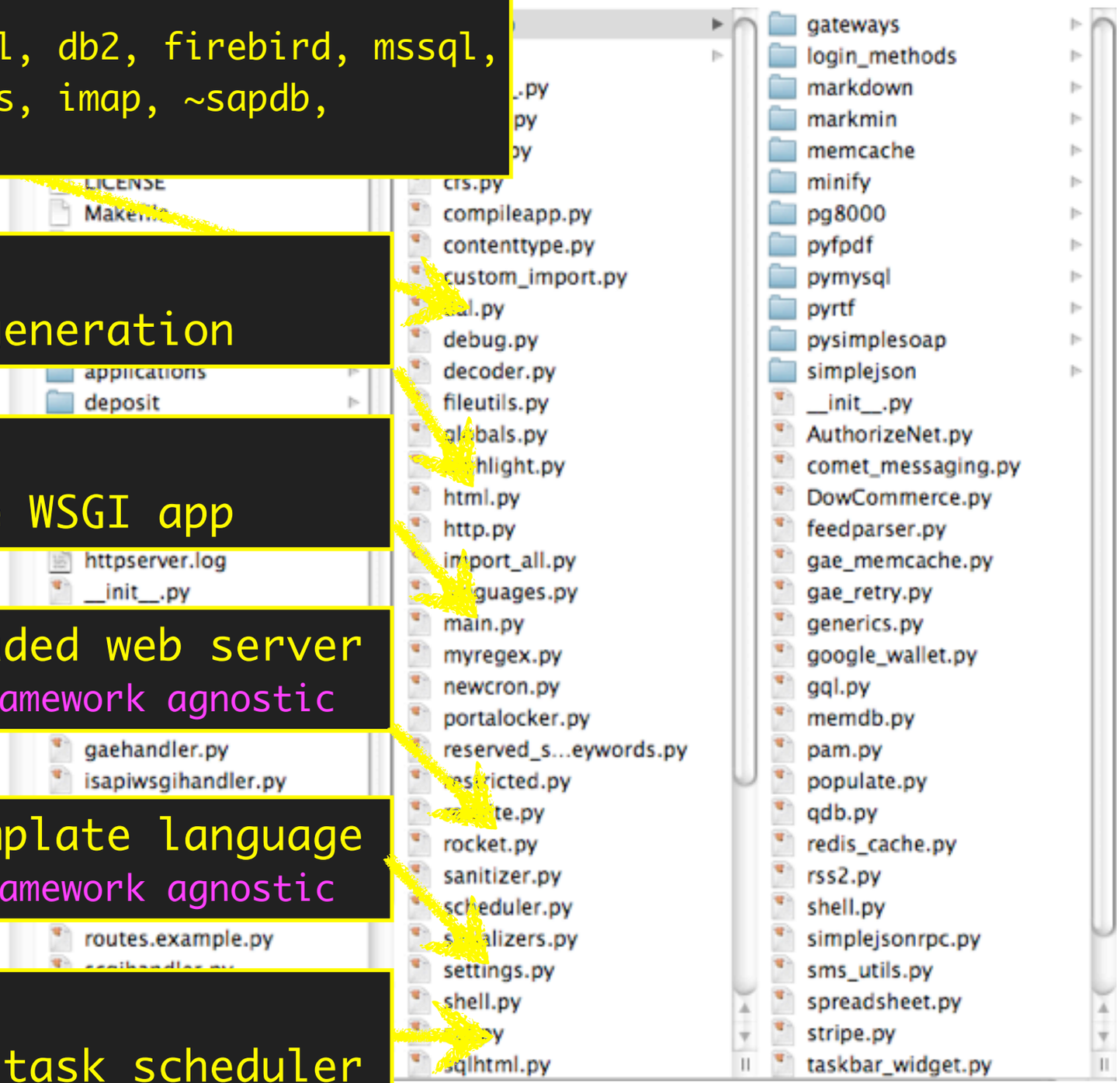
html.py & sqlhtml.py:
helpers, html parser, form generation

main.py:
contains "def wsgibase", the WSGI app

rocket.py: ssl-enabled threaded web server
single file / no dependencies / framework agnostic

template.py: pure python template language
single file / no dependencies / framework agnostic

cron.py: pure python cron
scheduler.py: master/worker task scheduler



Document processing:
pyfpdf (html to pdf), pyrtf (generates Rich Text Format)
markdown, MARKMIN (like markdown but better ;-)

Protocols:
rss, feedparser, json, PySimpleSOAP, ...

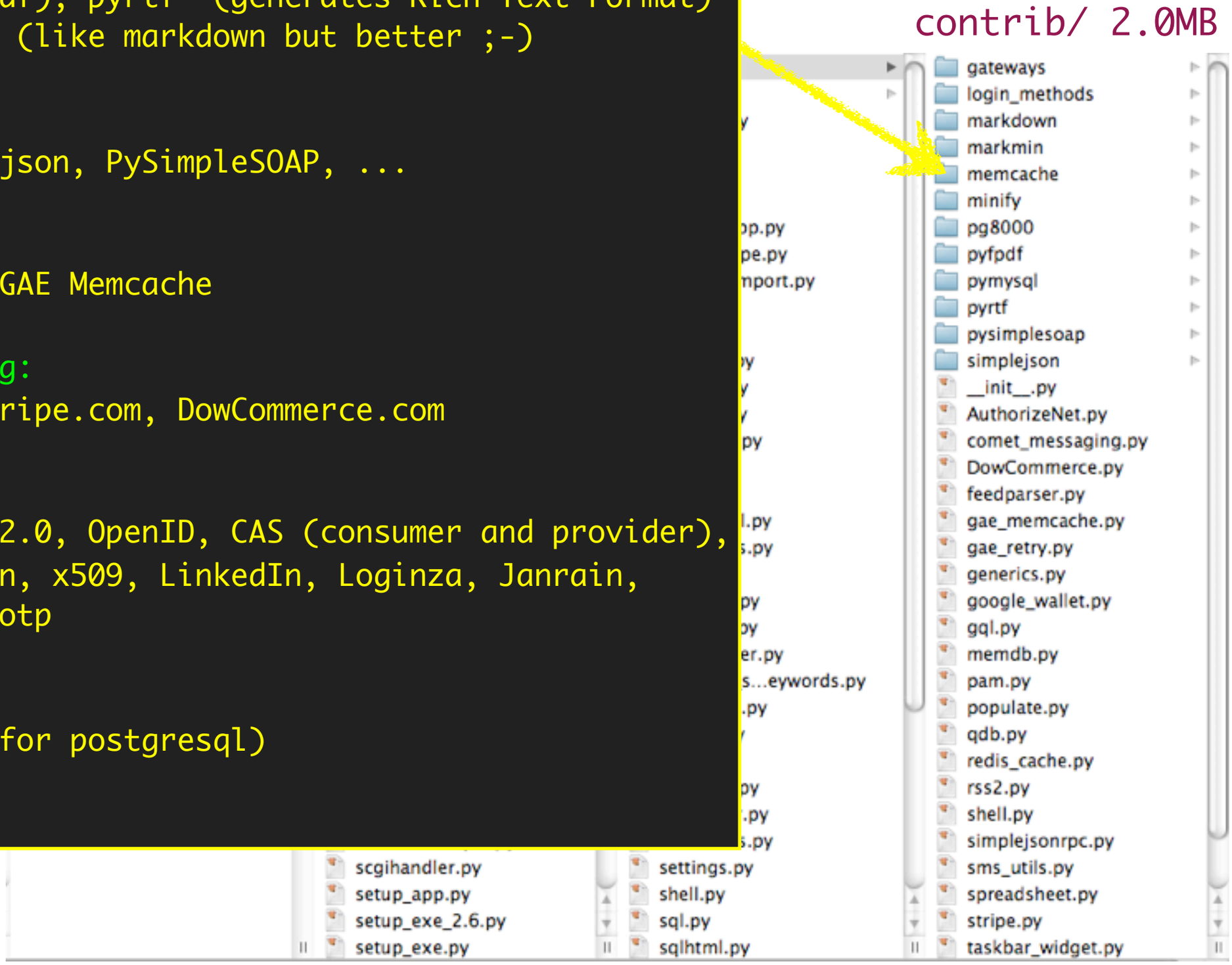
Caching Adapters:
Redis, Memcache, GAE Memcache

Payment processing:
Authorize.net, Stripe.com, DowCommerce.com

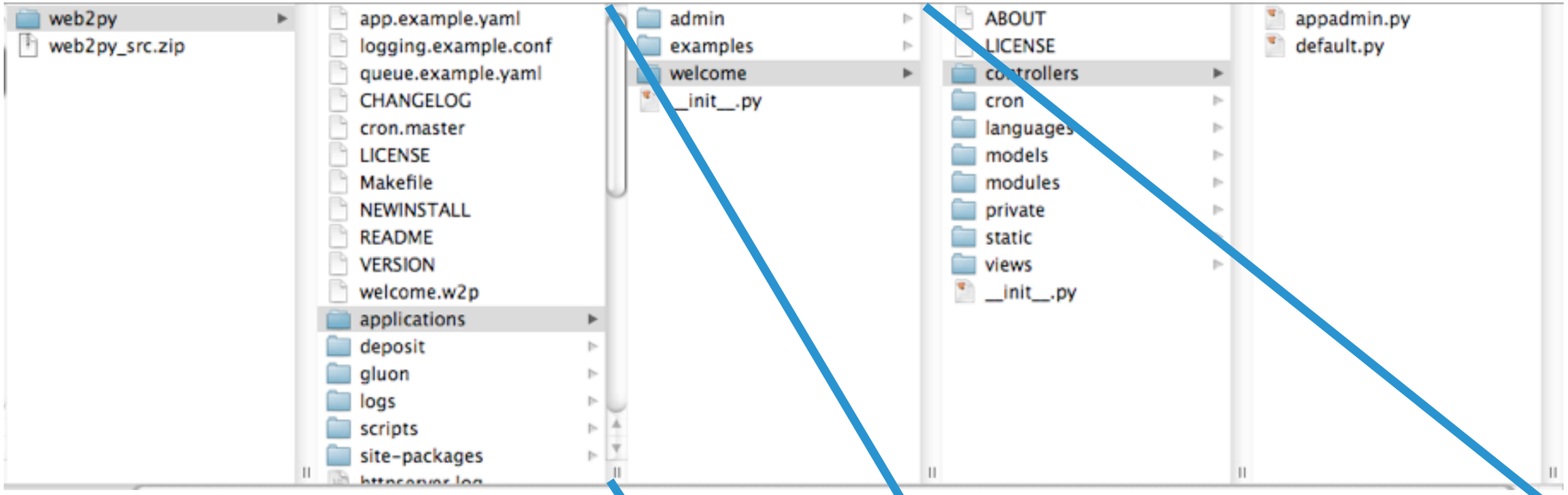
Login Methods:
Oauth 1.0, Oatuh 2.0, OpenID, CAS (consumer and provider),
PAM, LDAP, Janrain, x509, LinkedIn, Loginza, Janrain,
gmail, Dropbox, motp

Drivers:
pyMySQL, pg8000 (for postgresql)

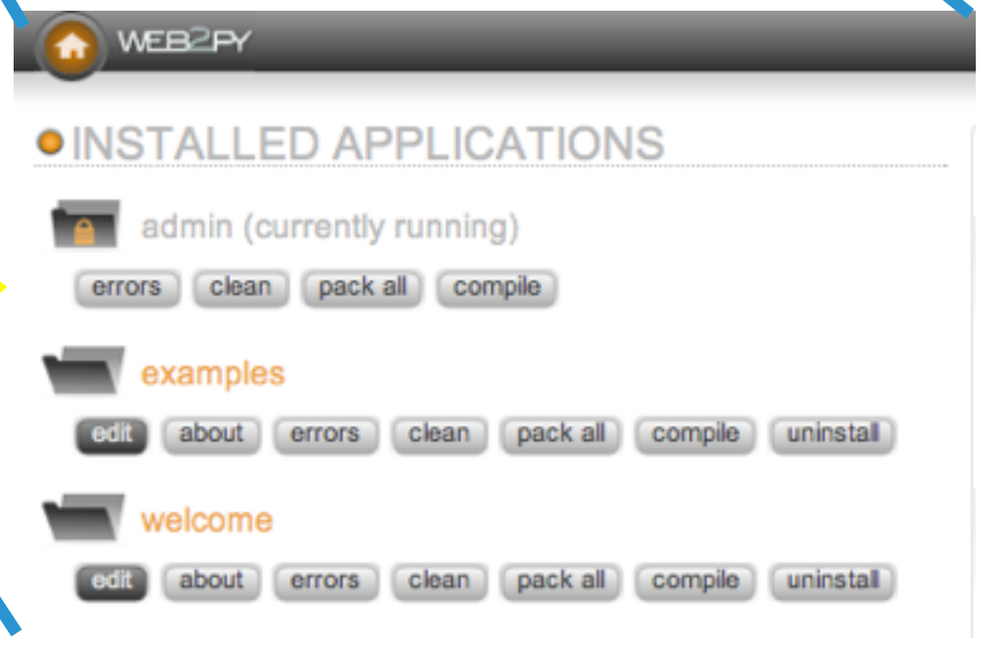
other stuff...



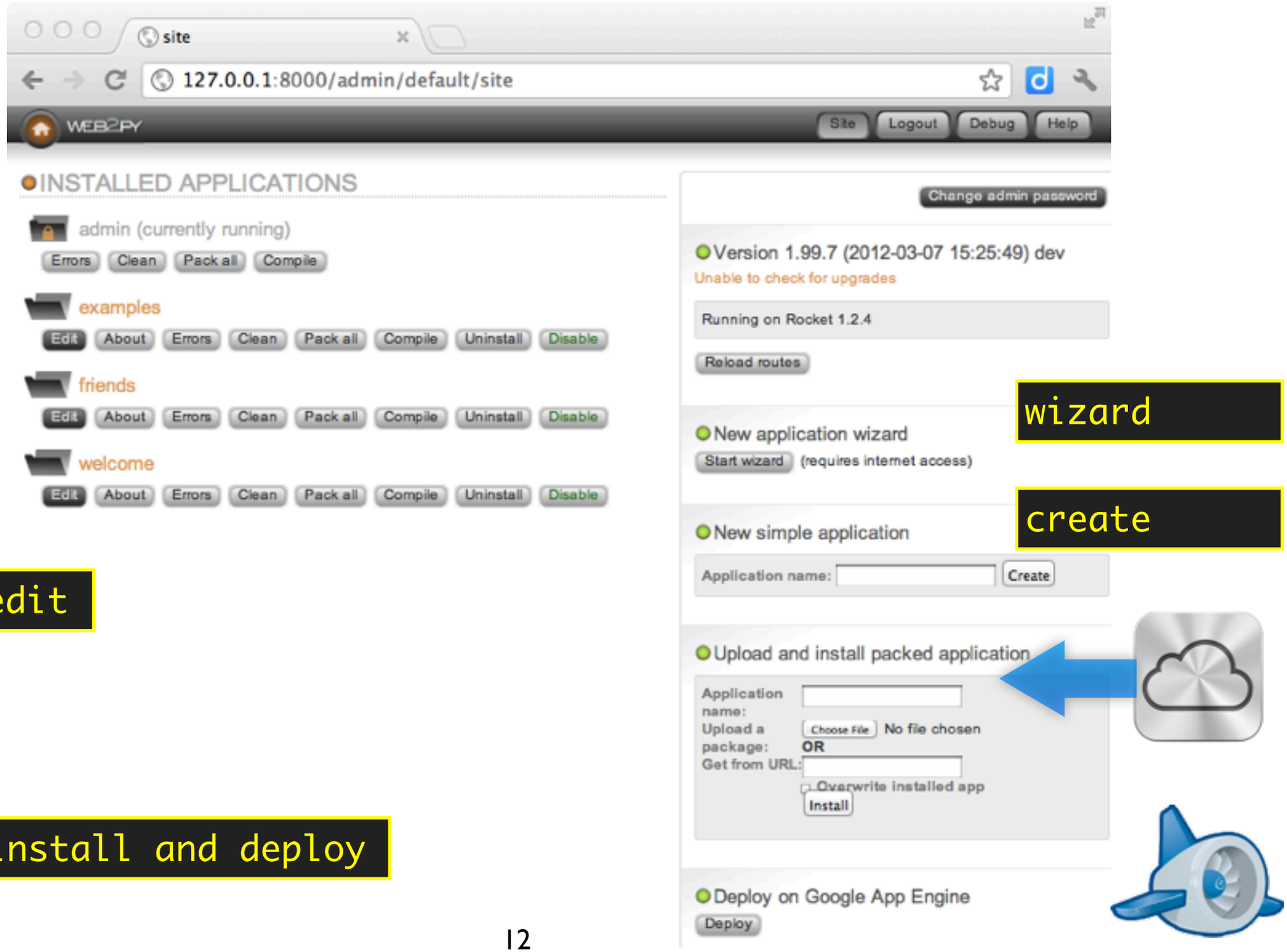
Multi-Project




```
admin:     the web based IDE
examples:  clone of the web site
welcome:   the scalffolding app
...        one web2py can run
           many projects
           without conflicts
```



Web Based IDE



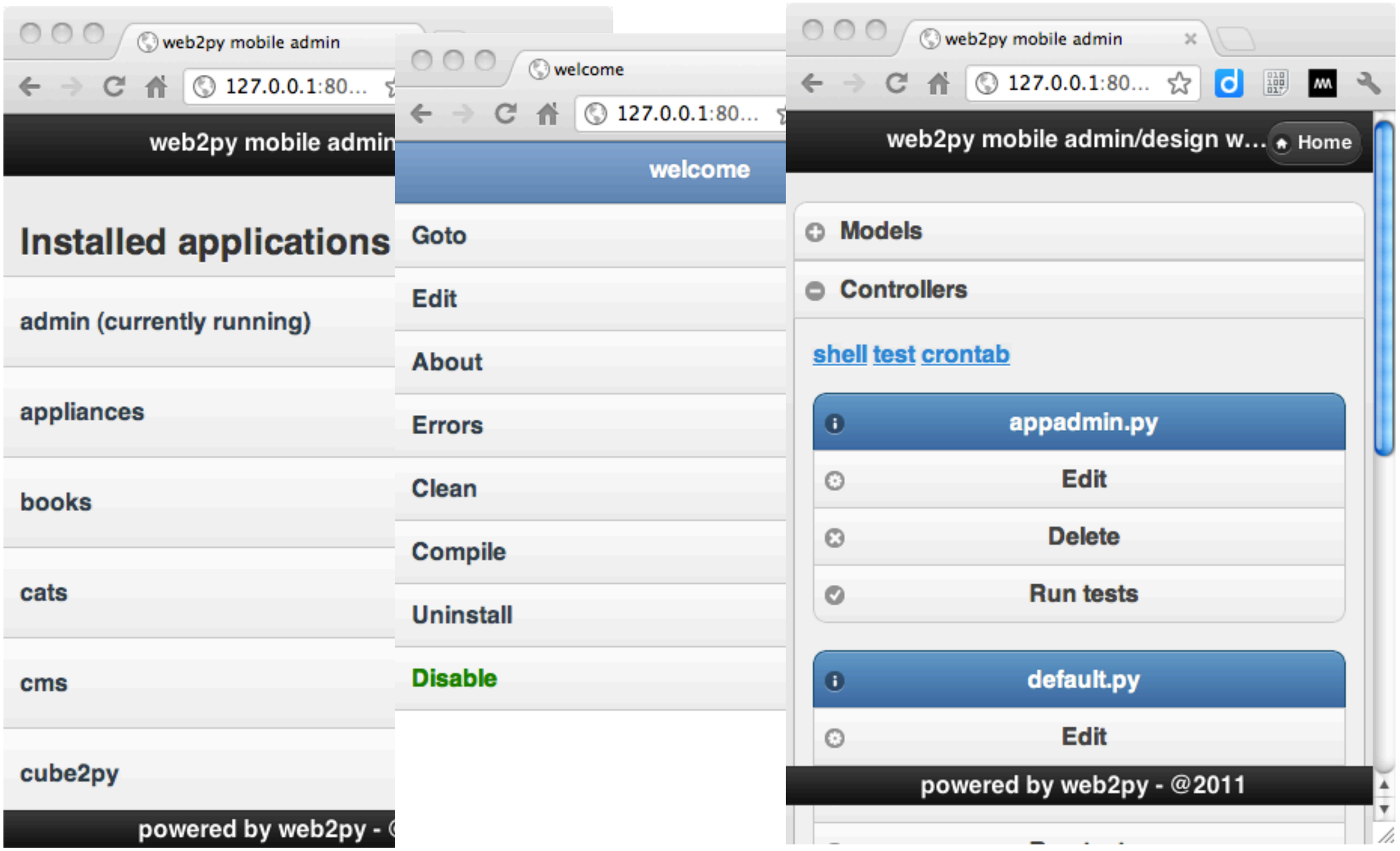
manage/edit

remote install and deploy

wizard

create

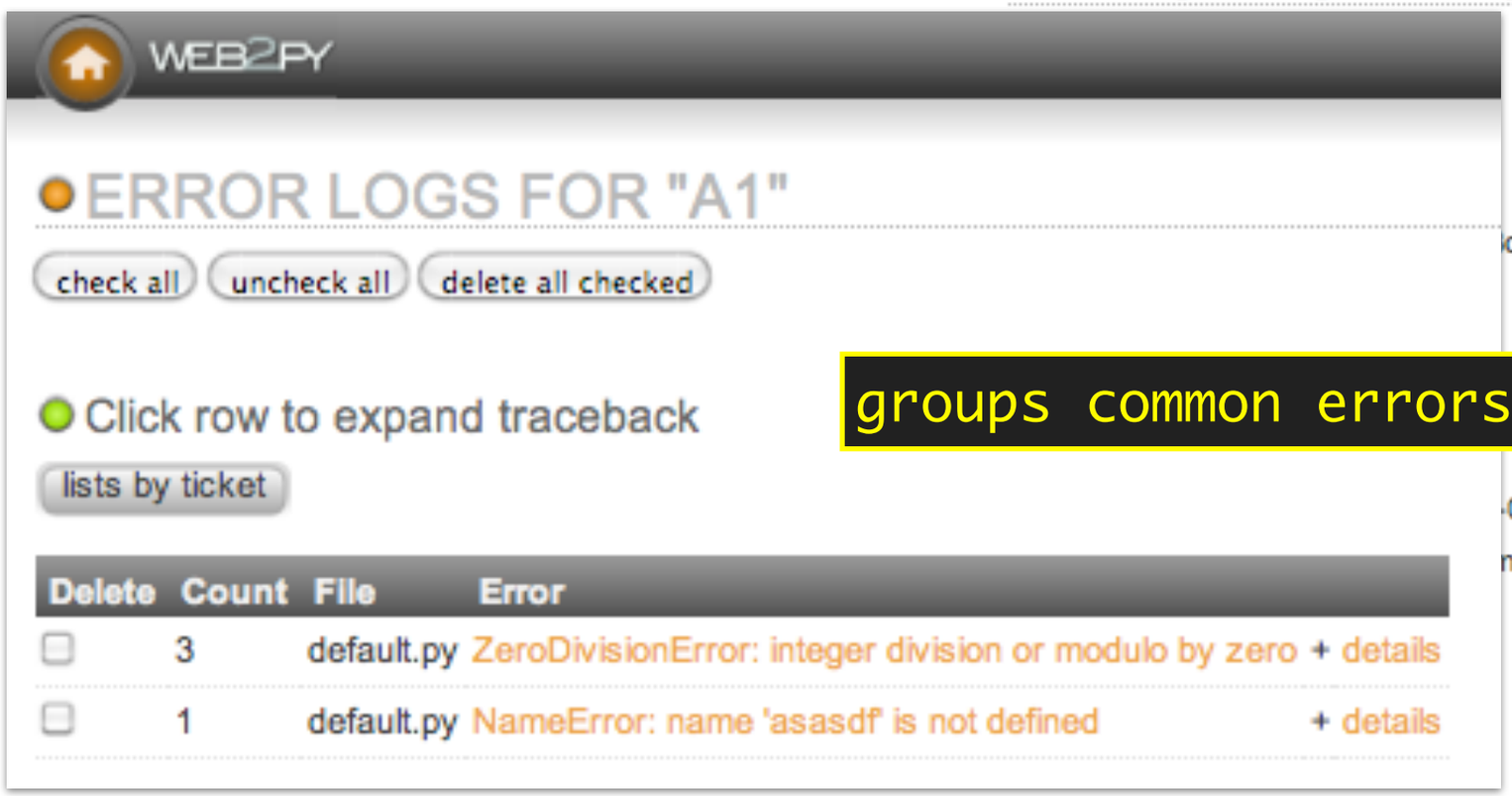
Mobile IDE


web2py mobile admin
127.0.0.1:80...
web2py mobile admin
Installed applications
admin (currently running)
appliances
books
cats
cms
cube2py
powered by web2py -
welcome
127.0.0.1:80...
welcome
Goto
Edit
About
Errors
Clean
Compile
Uninstall
Disable
web2py mobile admin
127.0.0.1:80...
web2py mobile admin/design w...
Home
Models
Controllers
shell test crontab
appadmin.py
Edit
Delete
Run tests
default.py
Edit
powered by web2py - @2011

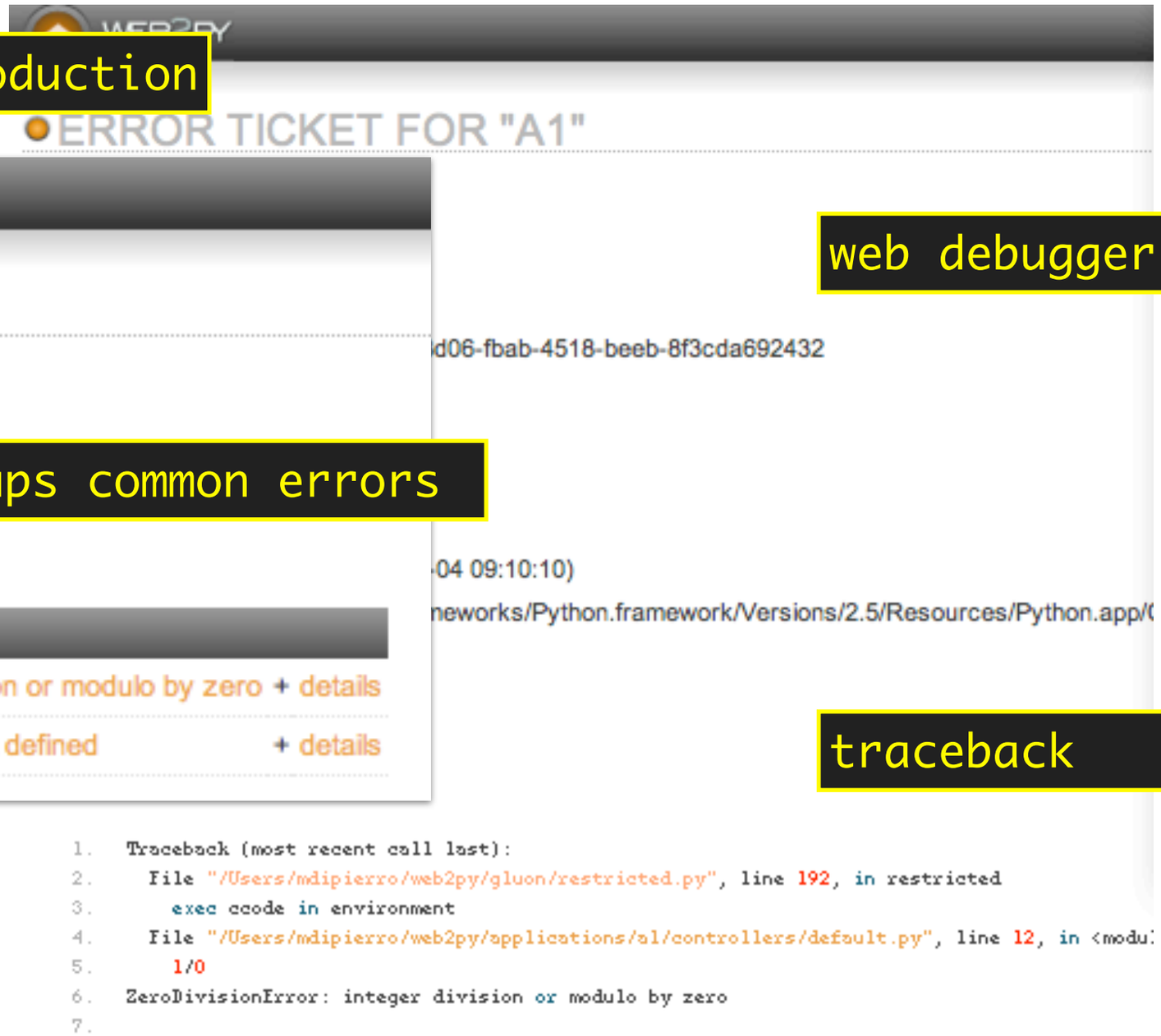
Web Translation

Tickets

no distiction debug vs production



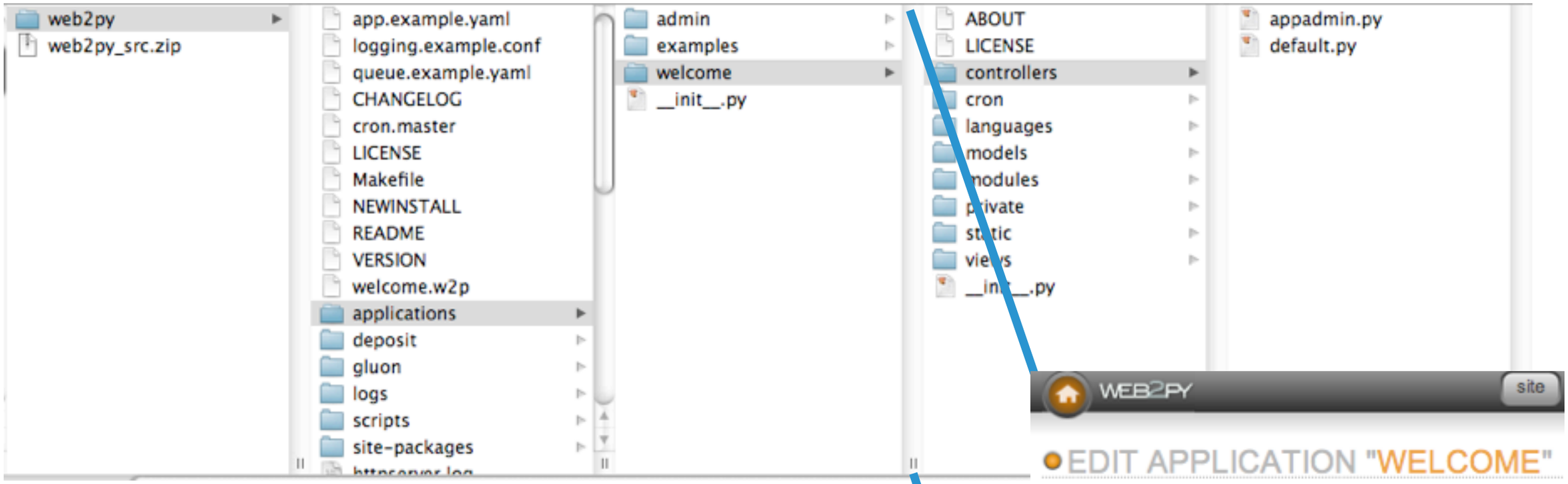
groups common errors



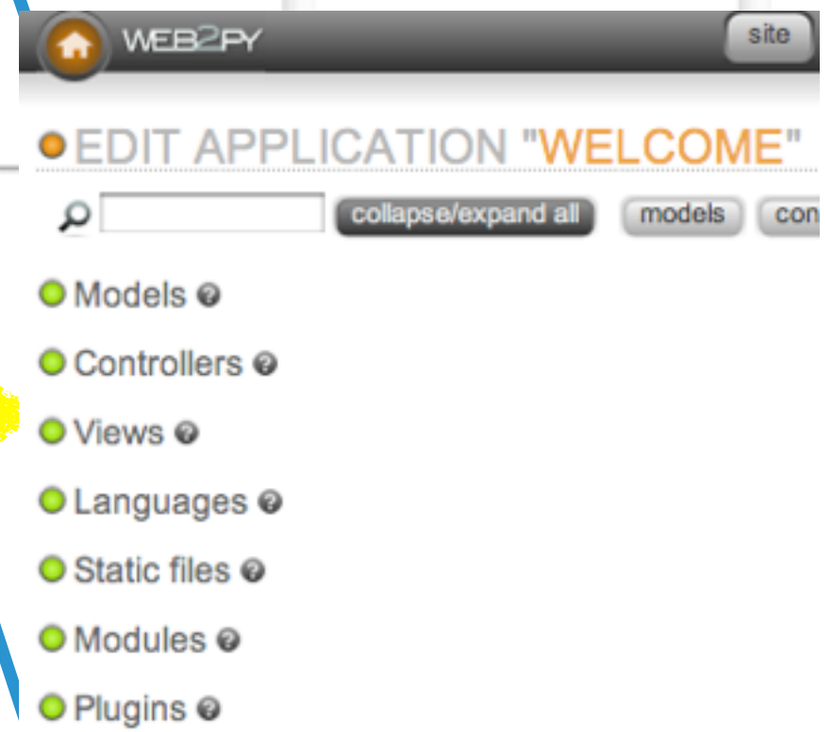
web debugger

traceback

Model-View-Controller




```
cd applications
mkdir friends
cp -r welcome/* friends
emacs friends/models/db.py
emacs friends/controllers/default.py
emacs friends/views/default/index.html
```



Let's build something

- URL shortening service
- Bookmarking service with tagging
- Click counter
- Url rating (WOT service)

models/myapp.py

```
Link = db.define_table(
    'link',
    Field('url',unique=True),
    Field('visits','integer',default=0),
    Field('screenshot','upload',writable=False),
    format = '%(url)s')

Bookmark = db.define_table(
    'bookmark',
    Field('link','reference link',writable=False),
    Field('category',
          requires=IS_IN_SET(['work','personal'])),
    Field('tags','list:string'),
    auth.signature)
```

controllers/default.py

```
def index():
    return dict()
```

models/myapp.py

```
def toCode(id):
    s,c = 'GKys67LJPDAFvcEp9rkwRd43fCjbxSXzMTQ28hgeUWuNYmqZt5VanBH',''
    while id: c,id = c+s[id % 55], id//55
    return c

def toInt(code):
    s,id = 'GKys67LJPDAFvcEp9rkwRd43fCjbxSXzMTQ28hgeUWuNYmqZt5VanBH',0
    for i in range(len(code)): id += s.find(code[i])*55**i
    return id

def shorten(id, row):
    s = URL('visit',args=toCode(id),scheme=True)
    return A(s,_href=s)

Bookmark.link.represent = shorten
Bookmark.id.readable = False
Bookmark.is_active.readable = False
Bookmark.is_active.writable = False
```

models/myapp_utils.py

```
from urllib import urlopen

def thumbalizr(url):
    """ free service to download web page thumbnail """
    return Bookmark.screenshot.store(
     urlopen('http://api1.thumbalizr.com/?url=%s&width=250'%url),
         filename = 'shot.png')

def wotrate(url):
    """ Web Of Trust service for web site rating """
    from gluon.tools import fetch
    from gluon.contrib.simplejson import loads
    wot = 'http://api.mywot.com/0.4/public_link_json?hosts=%s/'
    url = url.split('/')[2]
    return loads(urlopen(wot % url).read())[url]
```

models/menu.py

```
response.menu = [
    (T('Home'), False, URL('default','index')),
    (T('My Bookmarks'), False, URL('default','bookmarks')),
    ]
```

controllers/default.py

http://127.0.0.1:8000/myapp/default/bookmark?url=http://www.google.com

```
@auth.requires_login()
def bookmark():
    """ allows users to bookmark a page an get short url """
    url = request.vars.url
    link = Link(url=url) or Link.insert(url=url)
    link.update_record(screenshot=thumbalizr(url))
    bid = Bookmark(link=link) or Bookmark.insert(link=link)
    rating = cache.ram(link.url,lambda:wotrate(link.url),3600)
    form = SQLFORM(Bookmark,bid).process(next='bookmarks')
    return locals()

def visit():
    """ tracks visitors """
    link = Link(toInt(request.args(0)))
    link.update_record(visits=link.visits+1)
    redirect(link.url)
```

controllers/default.py

```
@auth.requires_login()
def bookmarks():
    """ allow users to manage their bookmarks """
    query = (Bookmark.created_by==auth.user.id)&\
            (Bookmark.link==Link.id)
    grid = SQLFORM.grid(query, create=False)
    return locals()

# stuff required for authenticatio
def user(): return dict(form=auth())
def download(): return response.download(request,db)
```

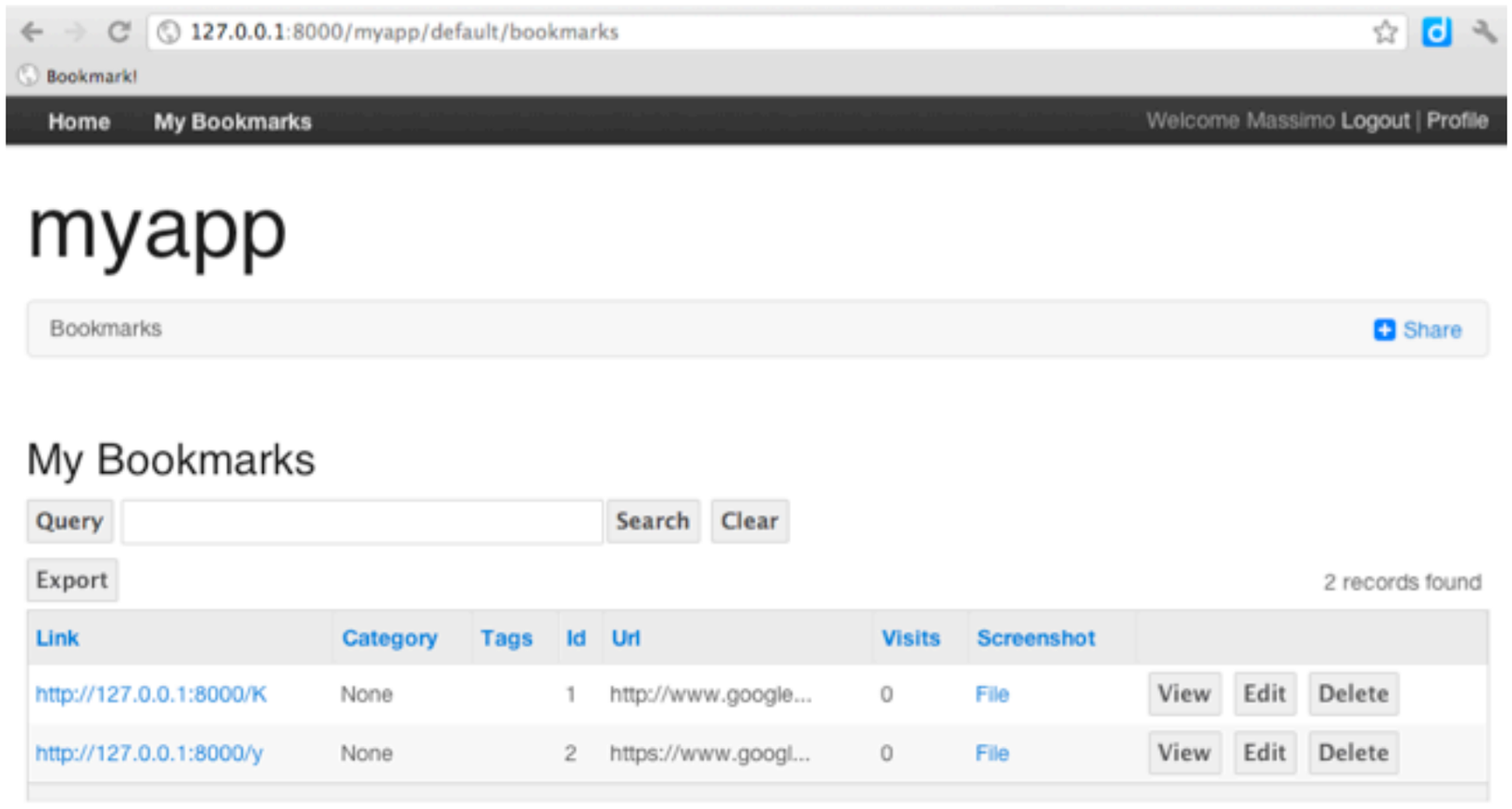
routes.py

localhost/K2 > localhost/myapp/default/visit/K2 > www.google.com

```
routes_in = [
    ('/(?P<code>\w+)','/myapp/default/visit/\g<code>'),
    ]

routes_out = [
    ('/myapp/default/visit/$code','/$code'),
    ]

routes_onerror=[
    ('myapp/*', 'myapp/default/error'),
    ]
```

views/default/bookmarks.py

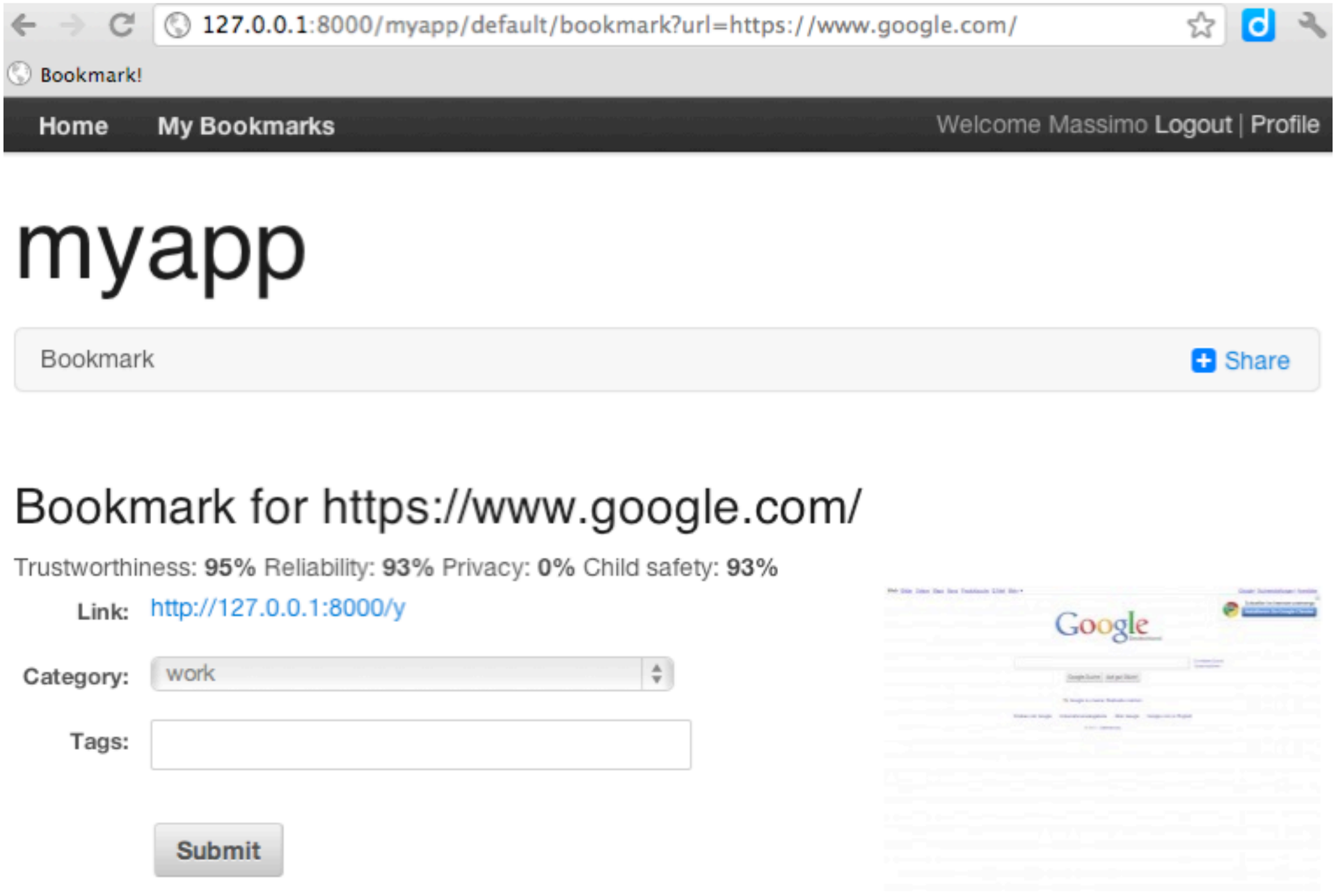
```
{{extend 'layout.html'}}
<h3>My Bookmarks</h3>
{{=grid}}
```



views/default/bookmark.py

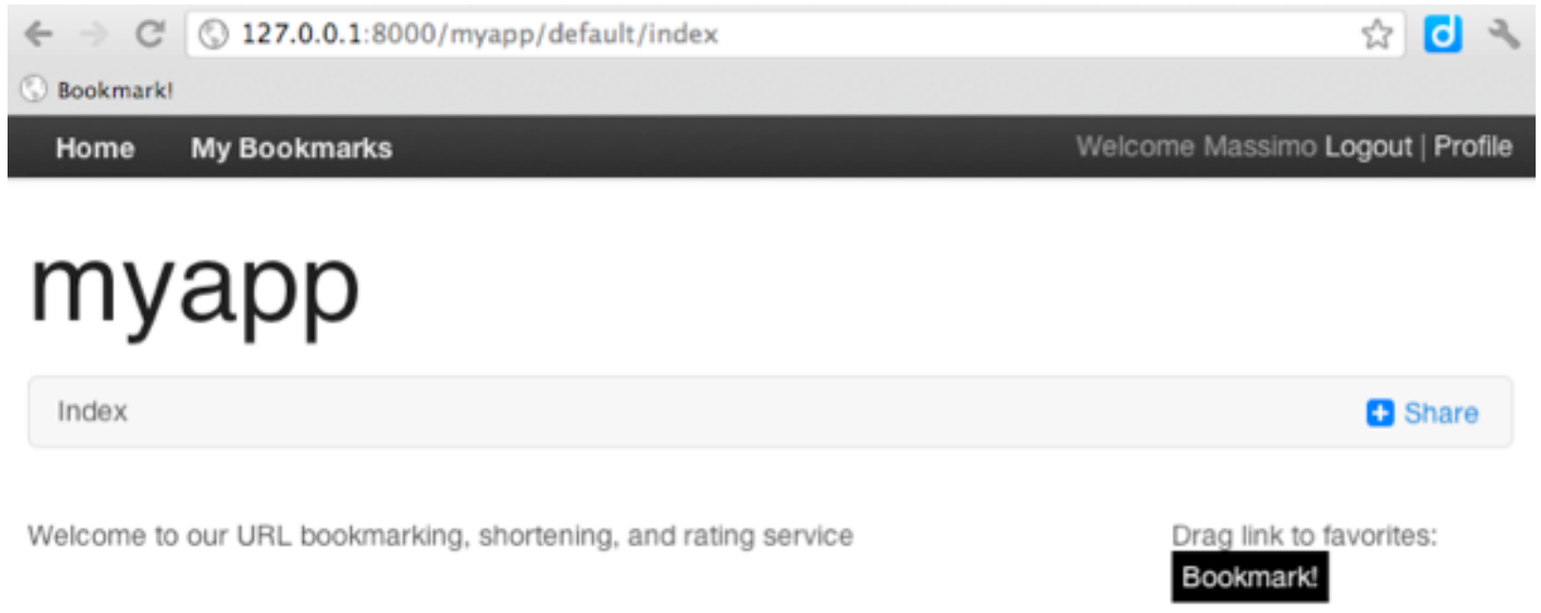
```
{{extend 'layout.html'}}

<h3>Bookmark for {{=url}}</h3>

{{def show(rating,k):}}
    <strong>{{=rating.get(k,[0])[0]}}%</strong>{{return}}
<div>
Trustworthiness: {{show(rating,'1')}}
Reliability: {{show(rating,'2')}}
Privacy: {{show(rating,'3')}}
Child safety: {{show(rating,'4')}}
</div>
<img src="{{=URL('default','download',
                 args=link.screenshot)}}" align="right"/>
{{=form}}
```

views/default/bookmark.py



views/default/index.html

```
{{right_sidebar_enabled=True}}
{{extend 'layout.html'}}

Welcome to our bookmarking & url shortening service

{{block right_sidebar}}
Drag link to favorites:
<a style="padding:5px;color:white;background-color:black;"
   title="Bookmark"
   href="javascript:window.location='{{=URL('bookmark',
   scheme=True)}}?url='+window.location">Bookmark!</a>
{{end}}
```



models/services.py

http://127.0.0.1:8000/myapp/services/call/xml/linksby?key=google
http://127.0.0.1:8000/myapp/services/api/google.json

```
@service.json
@service.xml
@service.jsonrpc
@service.xmlrpc
@service.amfrpc3('domain')
@service.soap()
def linksby(key=''):
    return db(Link.id==Bookmark.link)\
        (Bookmark.tags.contains(key)).select(Link.ALL,distinct=True)

@request.restful()
def api():
    def POST(**vars):
        raise HTTP(501)
    def GET(key=''):
        return dict(result=linksby(key).as_list())
    return locals()

def call(): return service()
```

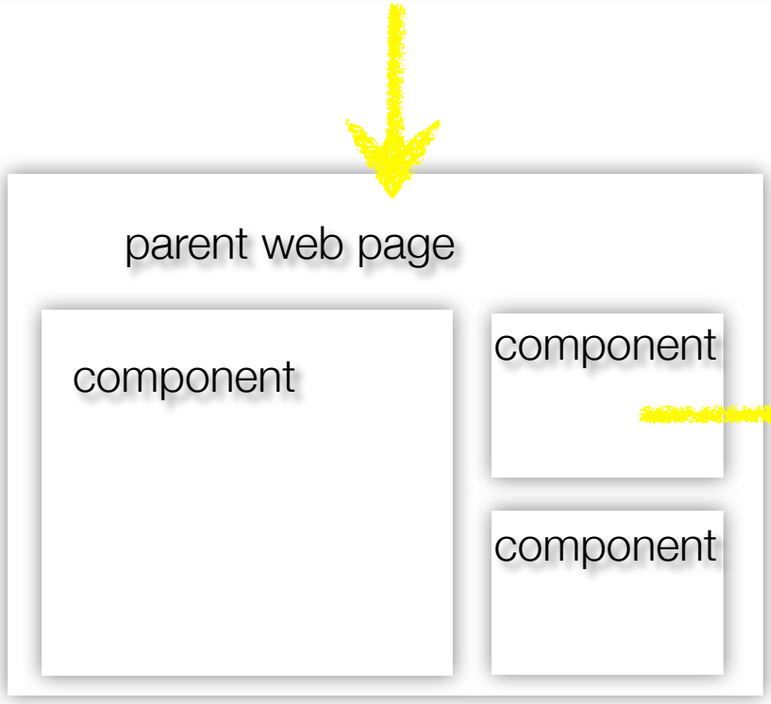
models/mytasks.py

```
from gluon.scheduler import Scheduler

def emailme():
    me = 'mdipierro@cs.depaul.edu'
    message = '%s users' % db(db.auth_user).count()
    mailer.send(to=me, subject='stats', message=message)
    return 'done!'

scheduler = Scheduler(db,dict(emailme=emailme))
if db(db.scheduler_task).isempty():
    db.scheduler_task.insert(
        application_name=request.application,
        task_name='email me user count',
        function_name='emailme',args=[],vars={},
        period=3600*24,repeats=0)
```

```
web2py.py -K friends &
web2py.py -K friends &
web2py.py -K friends &
```

Components

```
<div>
{{=LOAD('plugin','component1',user_signature=True,ajax=True)}}
{{=LOAD('plugin','component2',user_signature=True,ajax=True)}}
{{=LOAD('plugin','component3',user_signature=True,ajax=True)}}
</div>
```




```
@auth.requires_signature()
def component1():
    form = SQLFORM(db.friend)
    if form.process().accepted:
        response.flash = 'done!'
    return locals()
```

Thank you!

3700 registered users