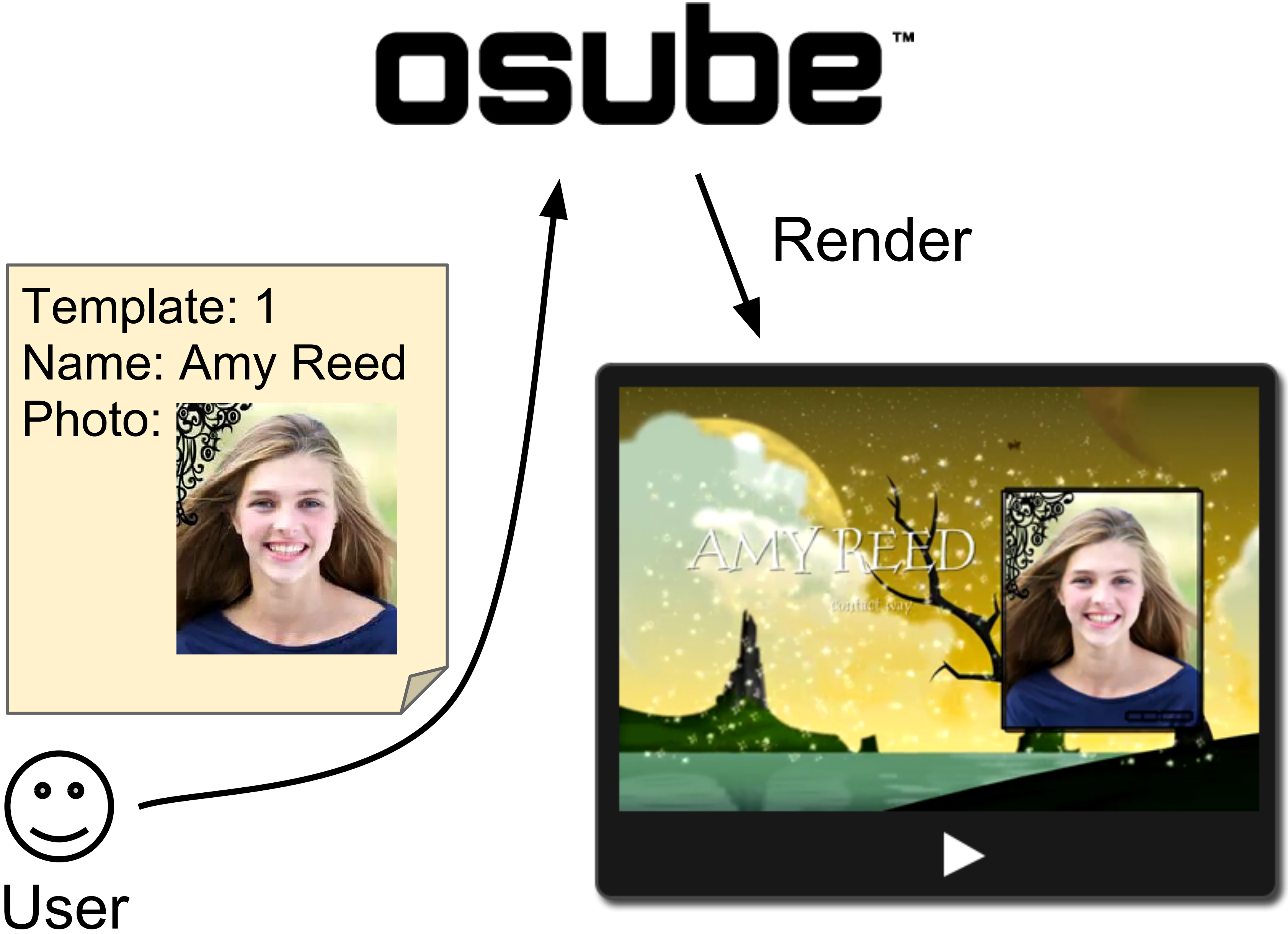
Building a Render Cloud

Eliang @ PyCon Taiwan 2013

About Me

梁長宏 (Eliang)

- Next Media Animation
- Osube
- The Manx Entertainment Group



osube™

Template: 1
Name: Amy Reed
Photo:

Render

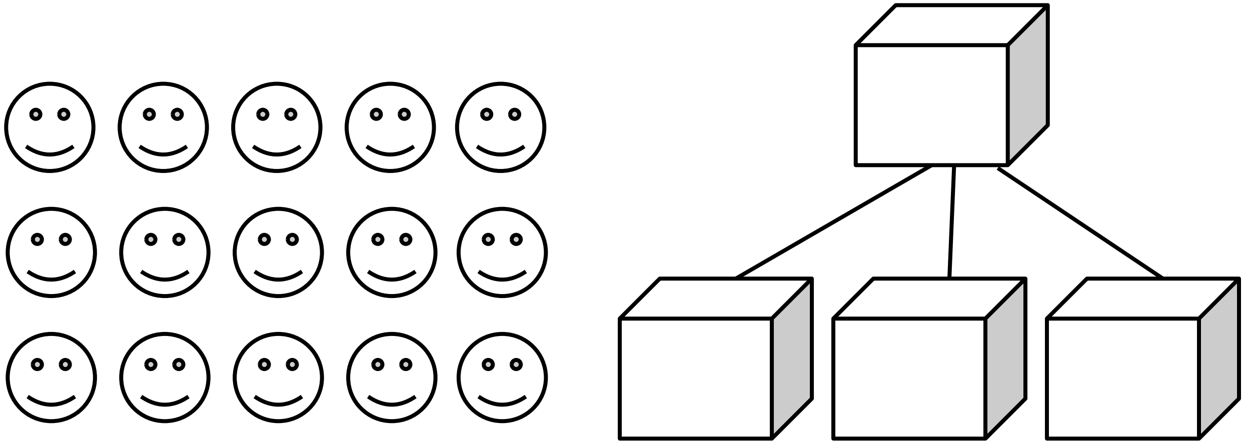
User

Tools We Used

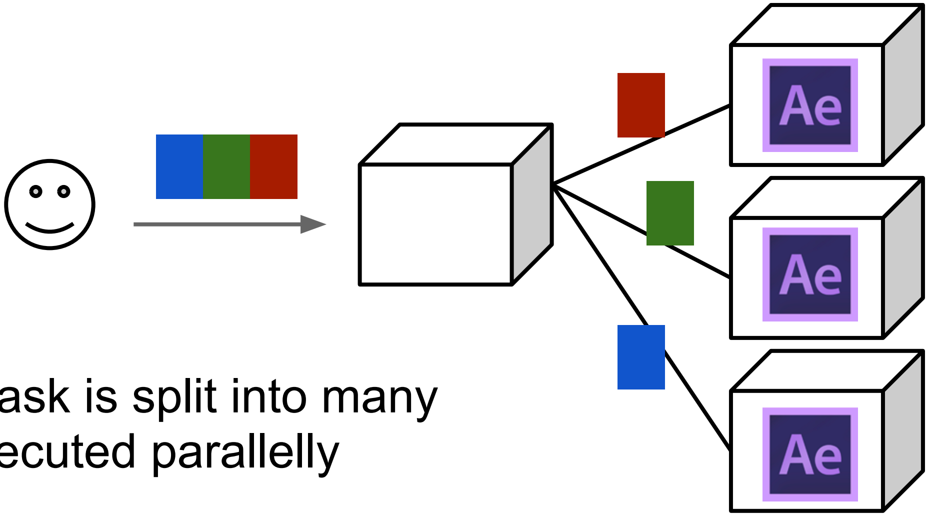
- Adobe After Effects
- FFmpeg
- RabbitMQ
- MySQL
- Python 2.7
- Django
- ...

Requirements

Scalable

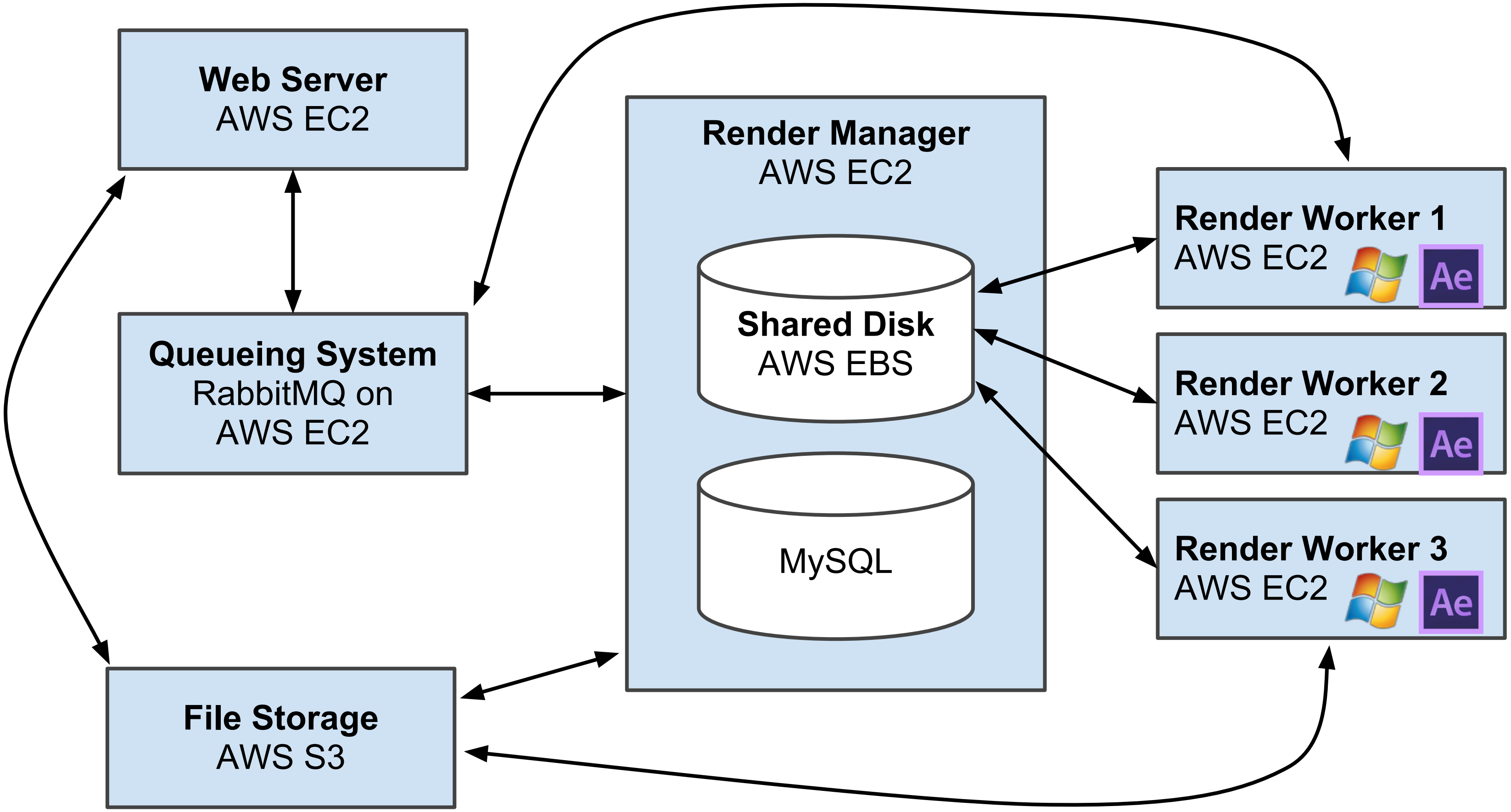


Parallel

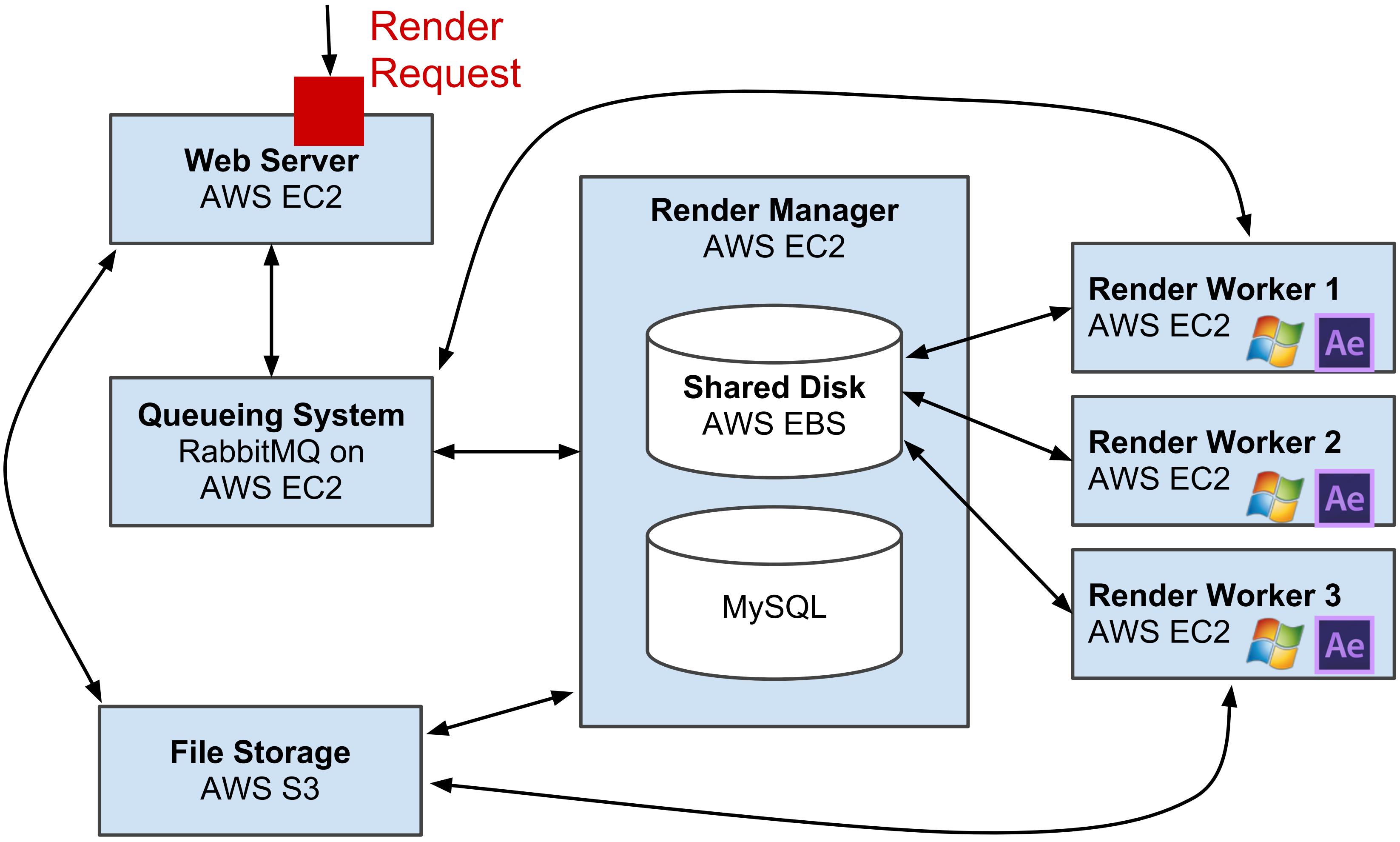


One render task is split into many subtasks, executed parallelly

Render Cloud *powered by* amazon web services



Render Cloud *powered by* amazon web services



Render Request

Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

File Storage
AWS S3

Render Manager
AWS EC2

Shared Disk
AWS EBS

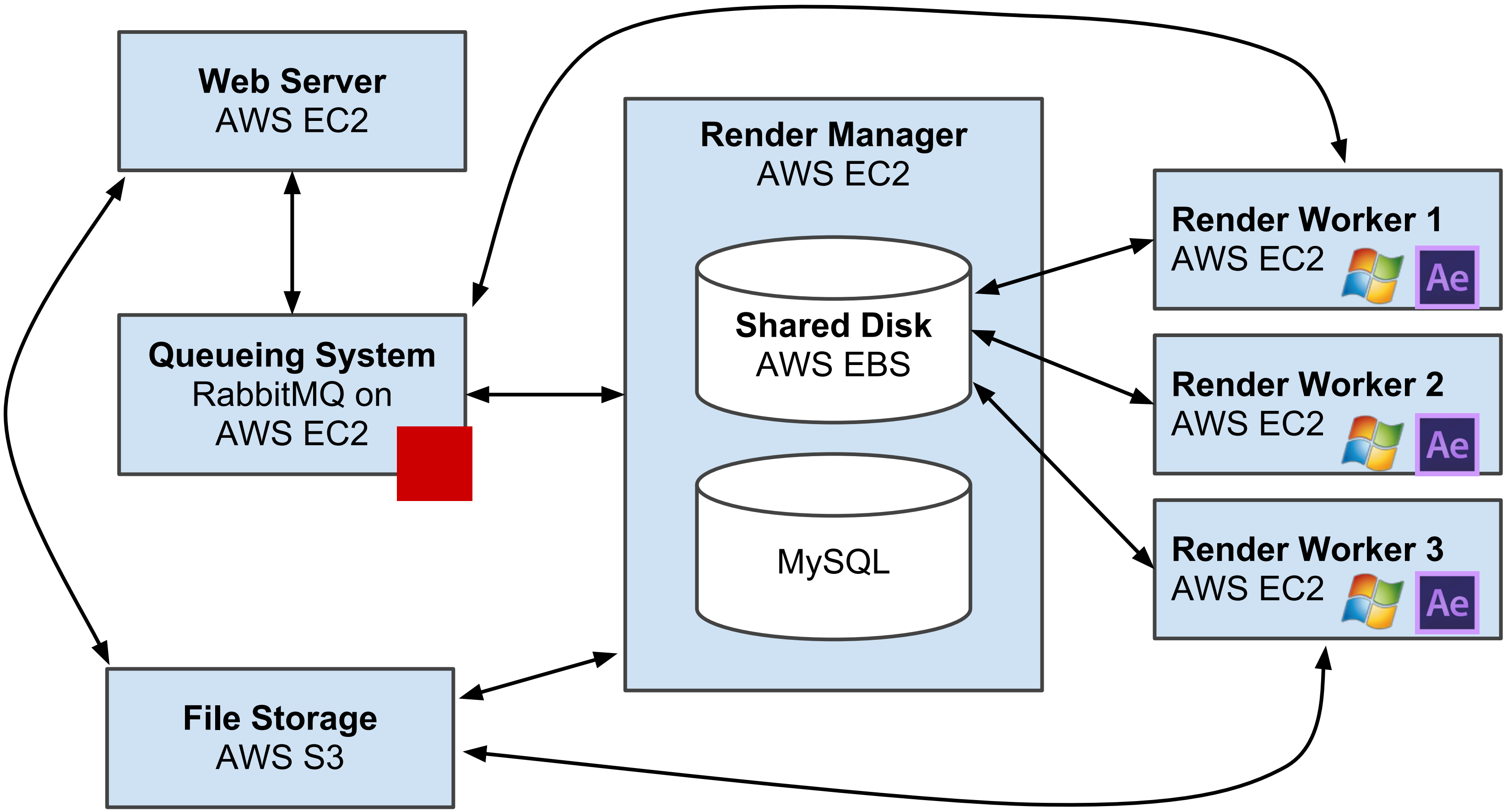
MySQL

Render Worker 1
AWS EC2

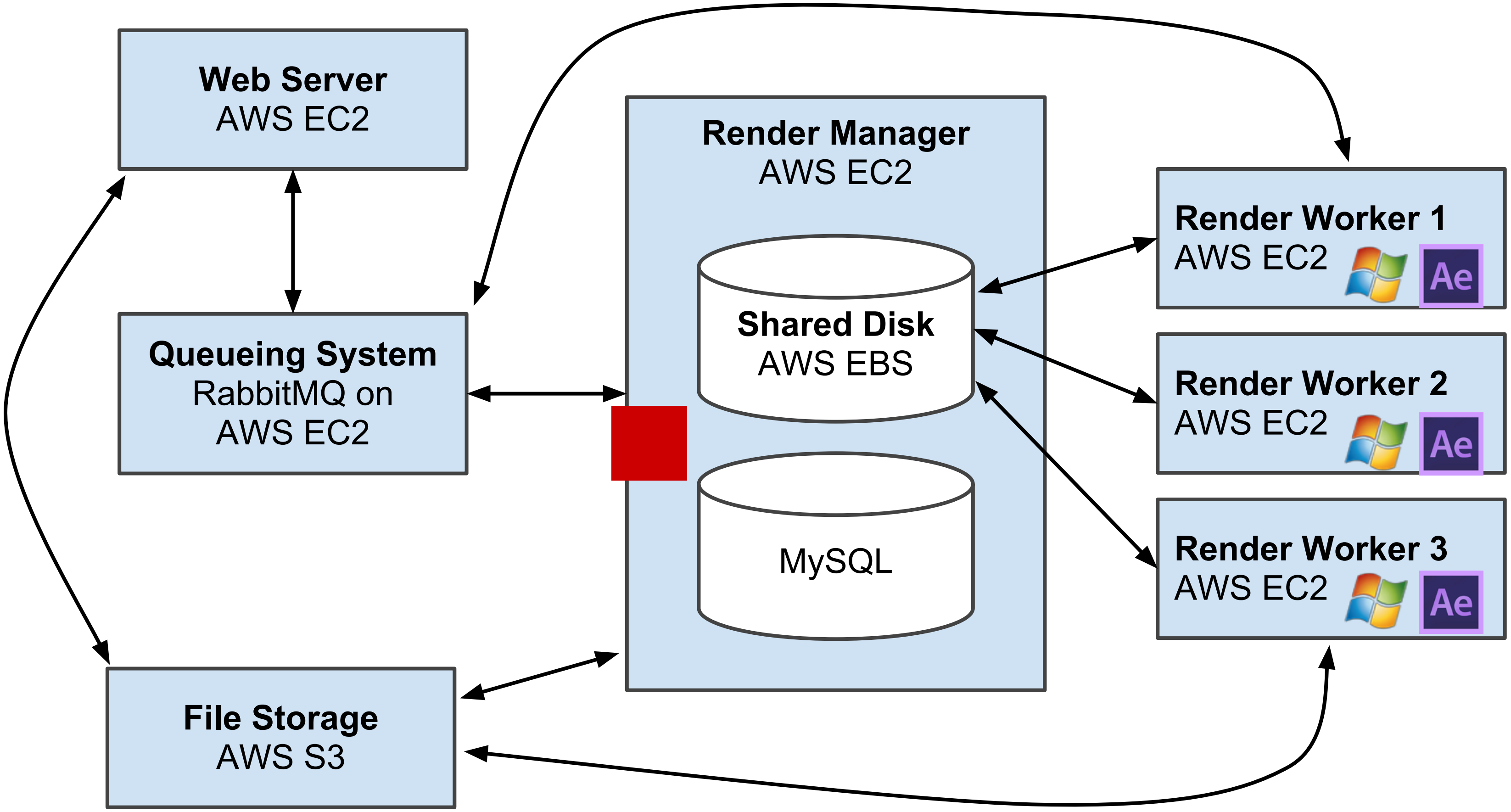
Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

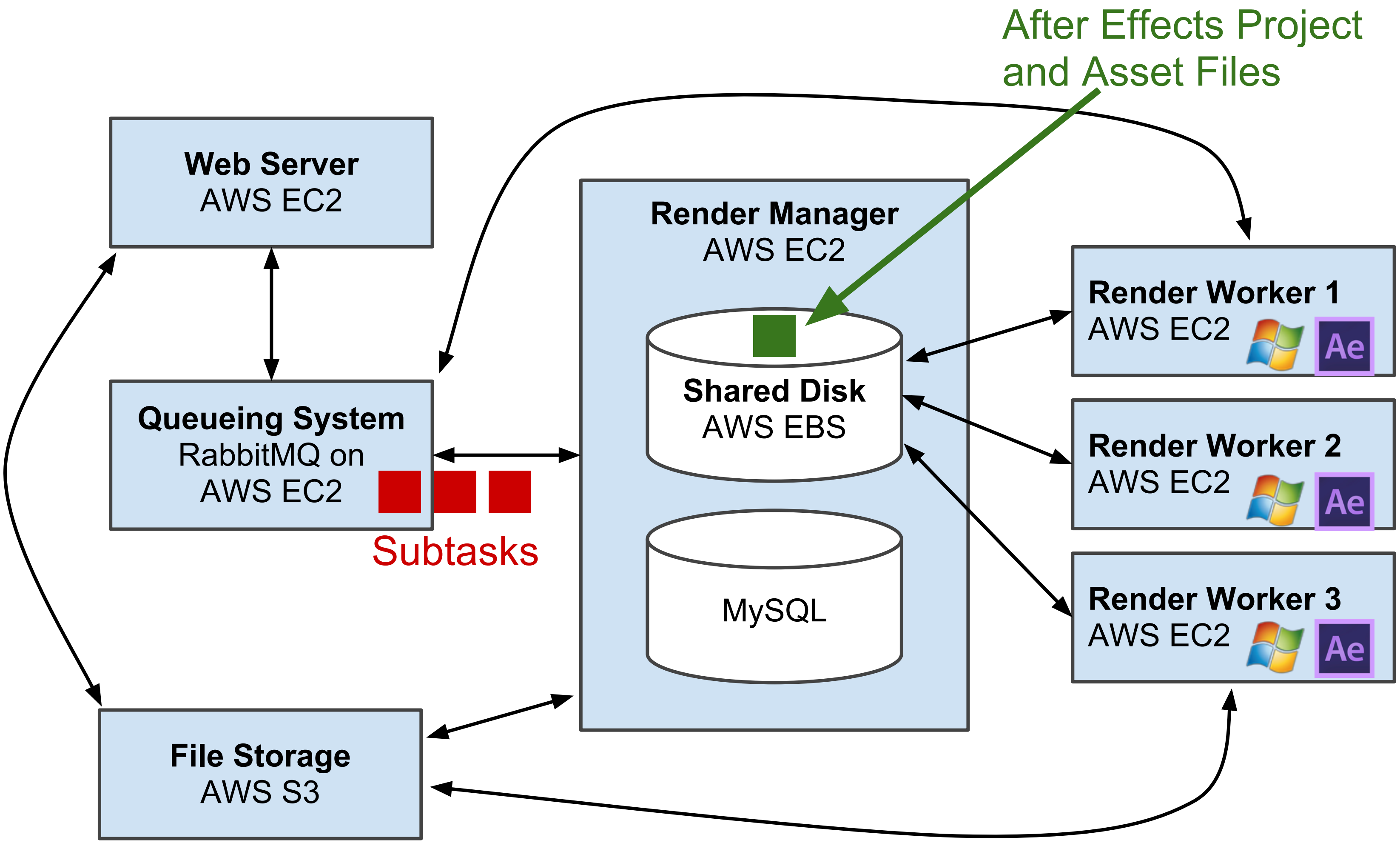
Render Cloud *powered by* amazon web services



Render Cloud *powered by* amazon web services



Render Cloud *powered by* amazon web services



After Effects Project and Asset Files

Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

Subtasks

Render Manager
AWS EC2

Shared Disk
AWS EBS

MySQL

Render Worker 1
AWS EC2

Ae

Render Worker 2
AWS EC2

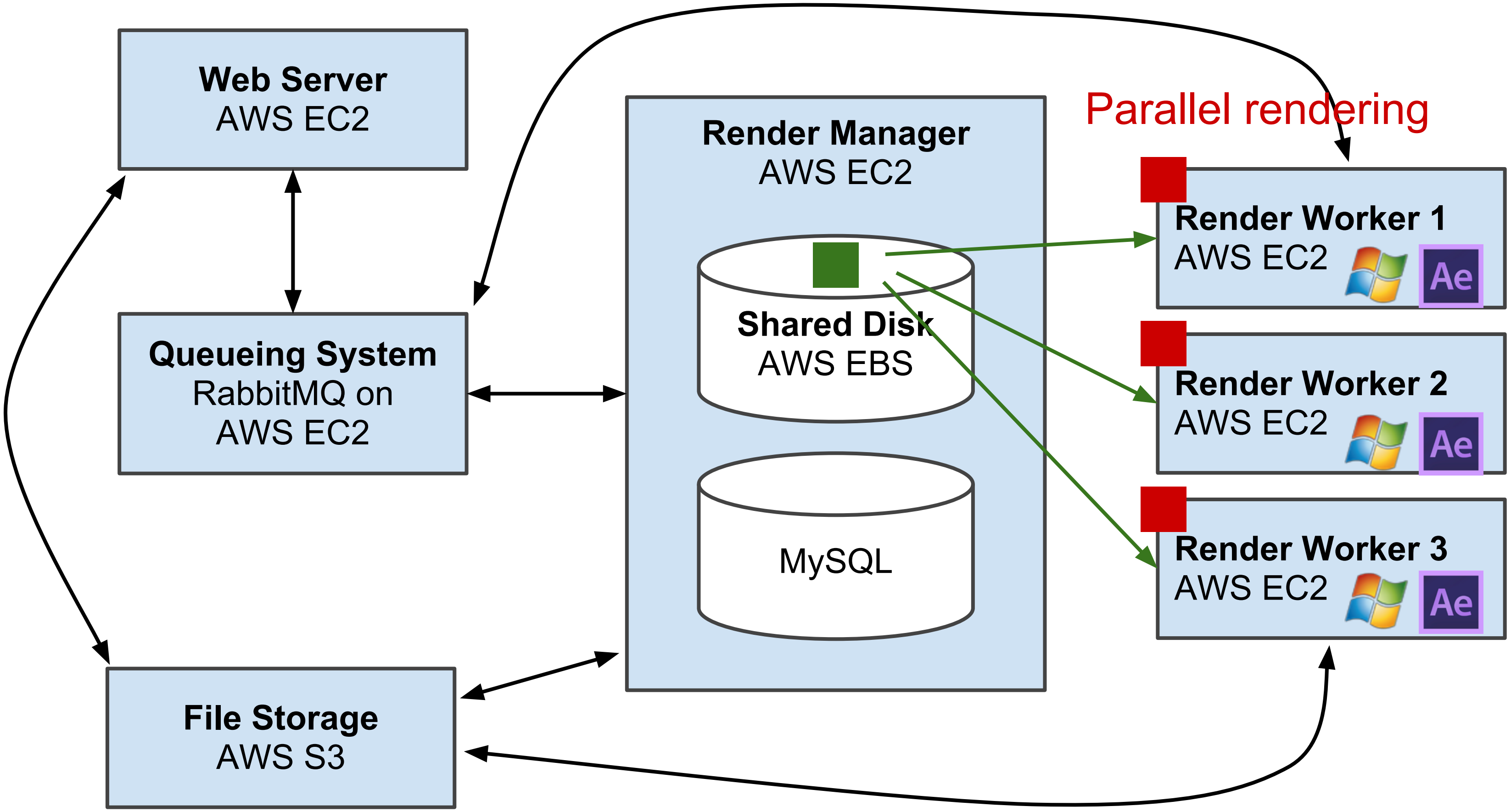
Ae

Render Worker 3
AWS EC2

Ae

File Storage
AWS S3

Render Cloud *powered by* amazon web services



Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

File Storage
AWS S3

Render Manager
AWS EC2

Shared Disk
AWS EBS

MySQL

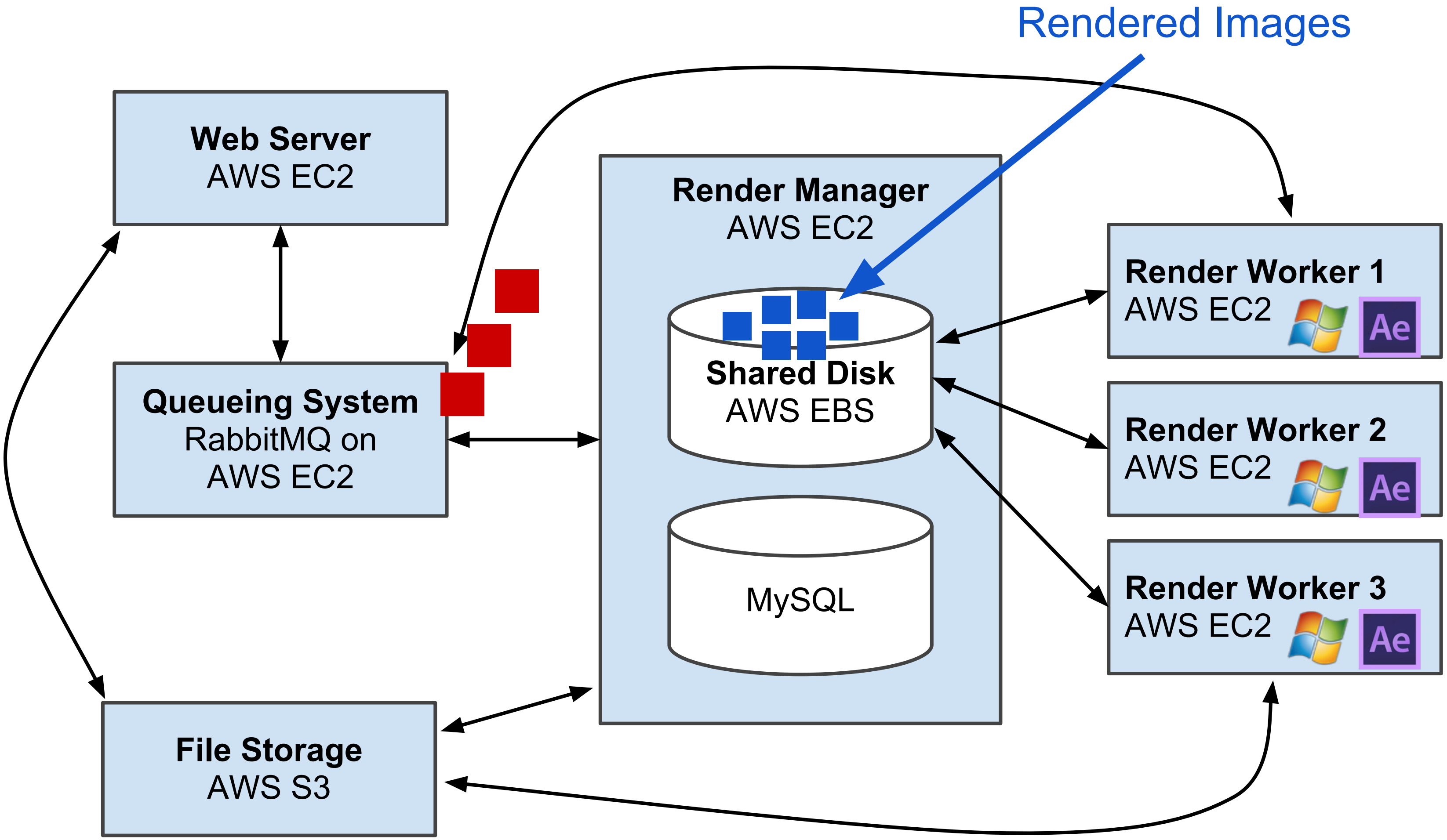
Parallel rendering

Render Worker 1
AWS EC2

Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

Render Cloud *powered by* amazon web services



Rendered Images

Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

File Storage
AWS S3

Render Manager
AWS EC2

Shared Disk
AWS EBS

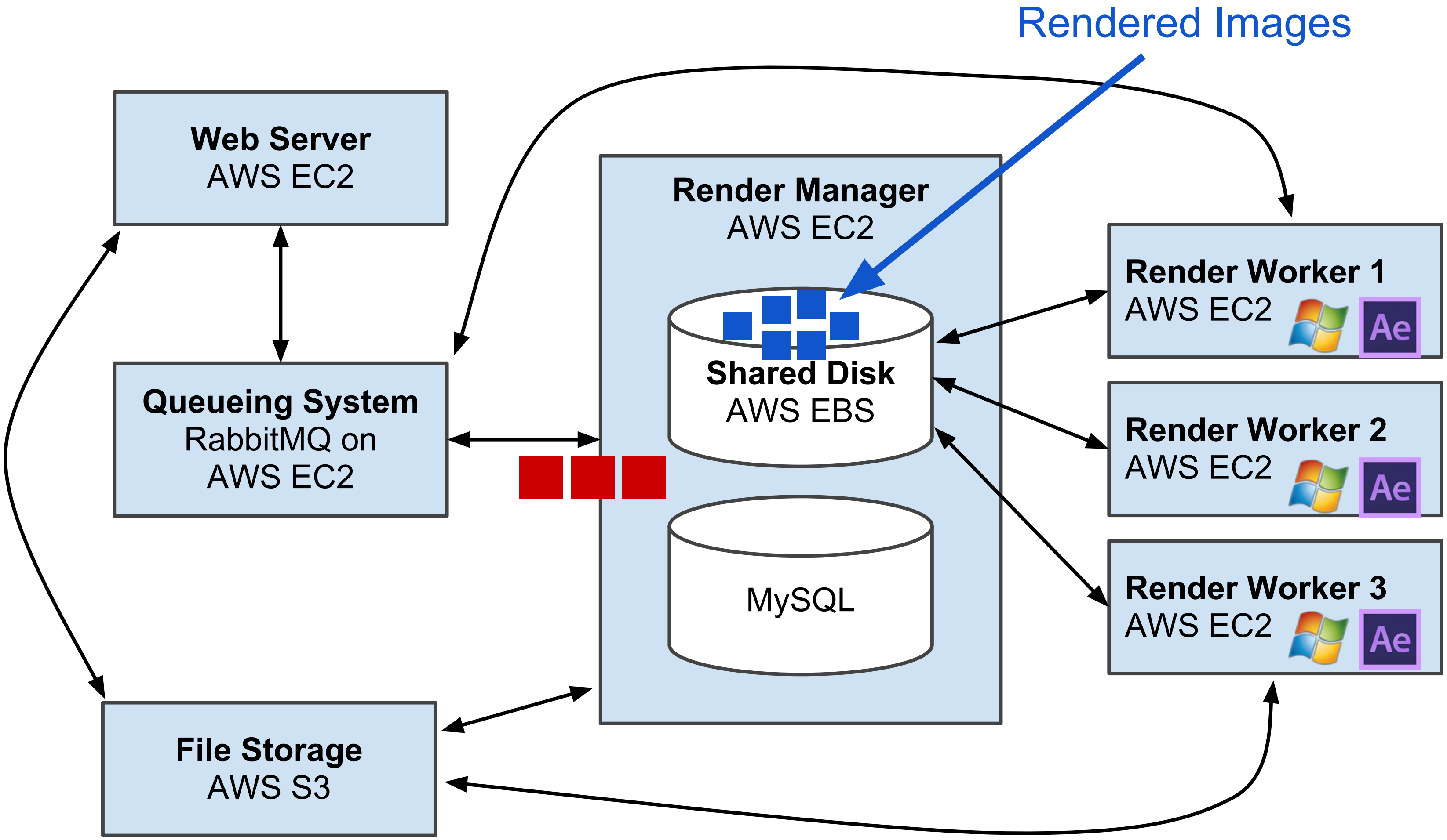
MySQL

Render Worker 1
AWS EC2

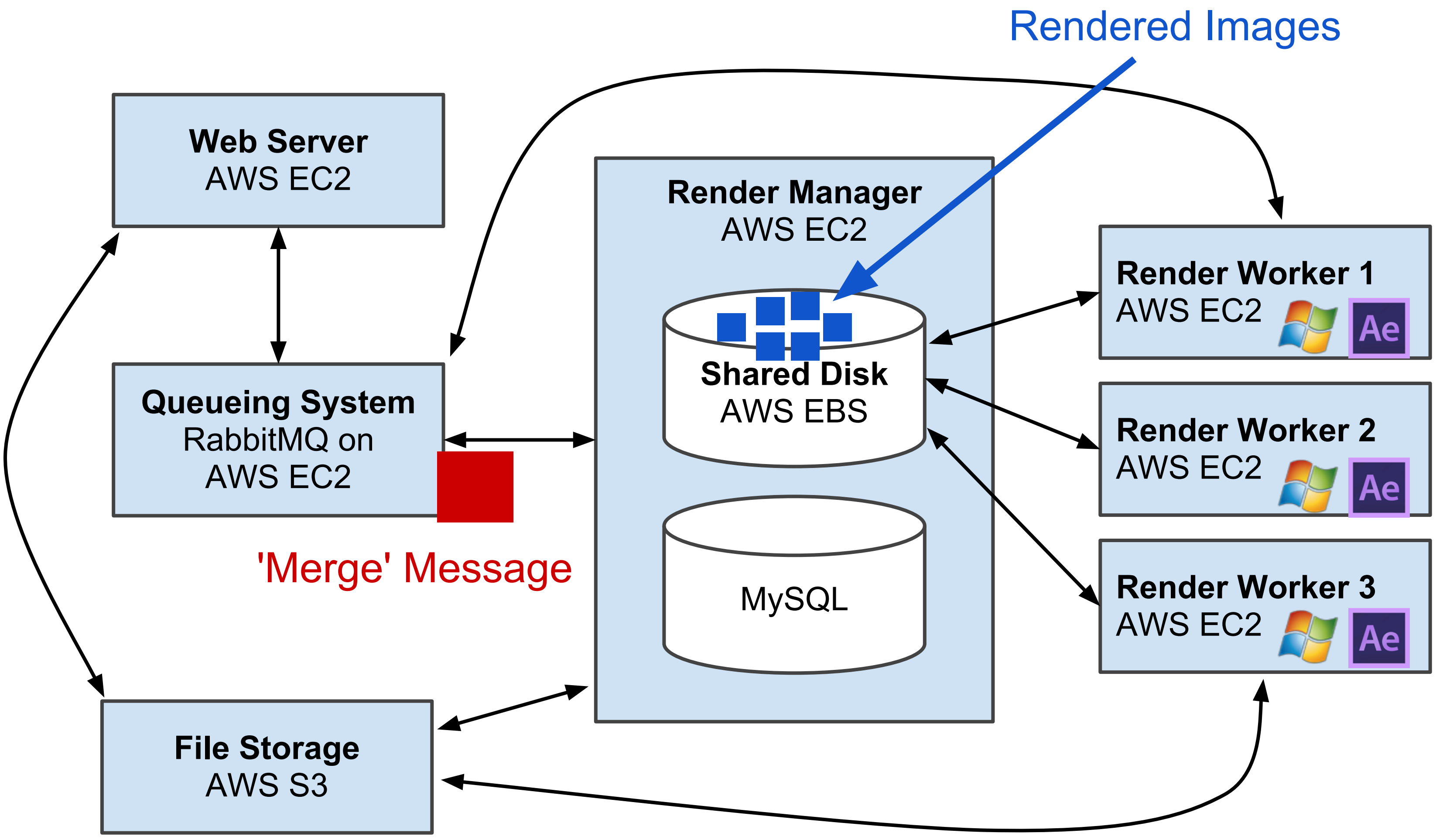
Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

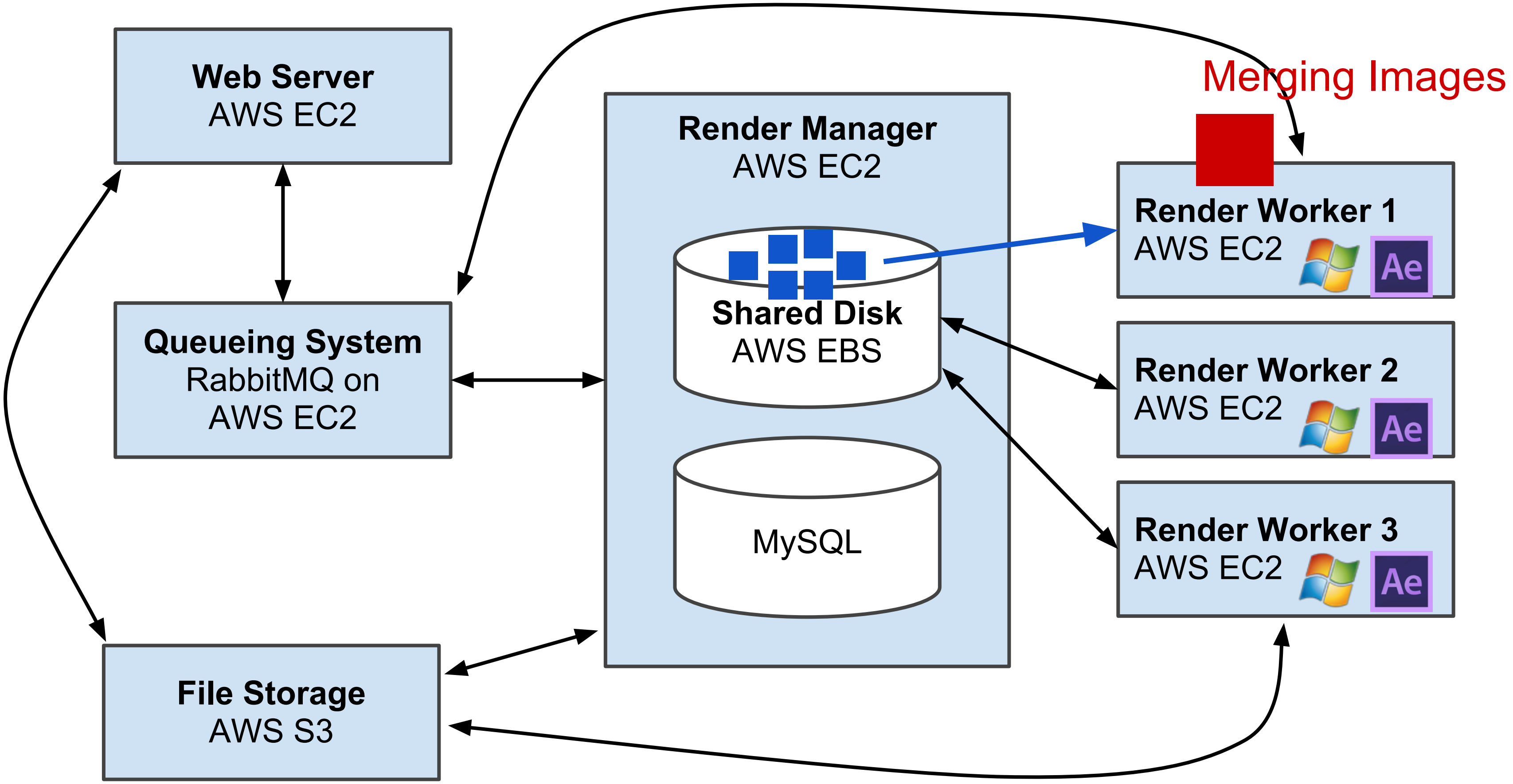
Render Cloud *powered by* amazon web services



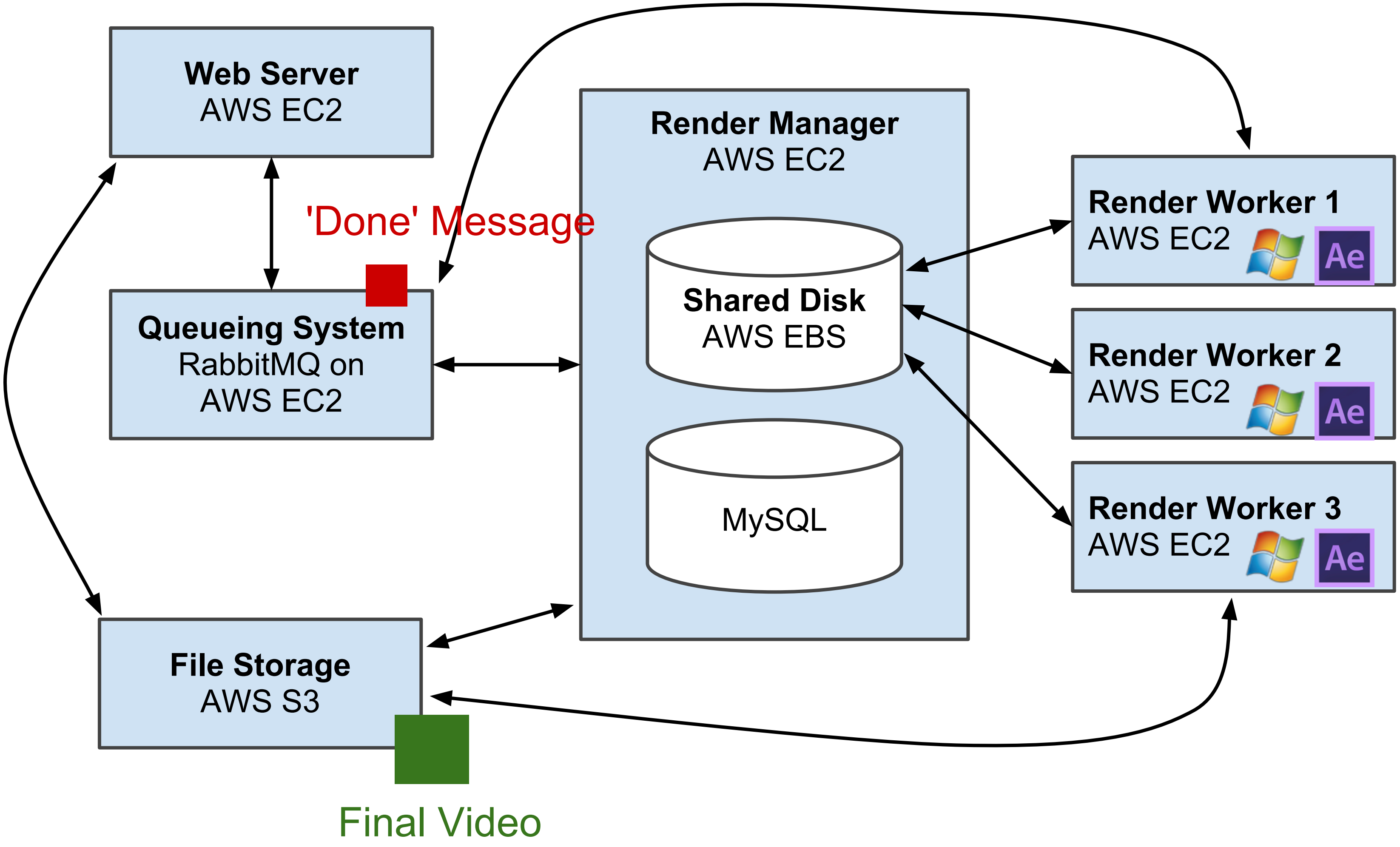
Render Cloud *powered by* amazon web services



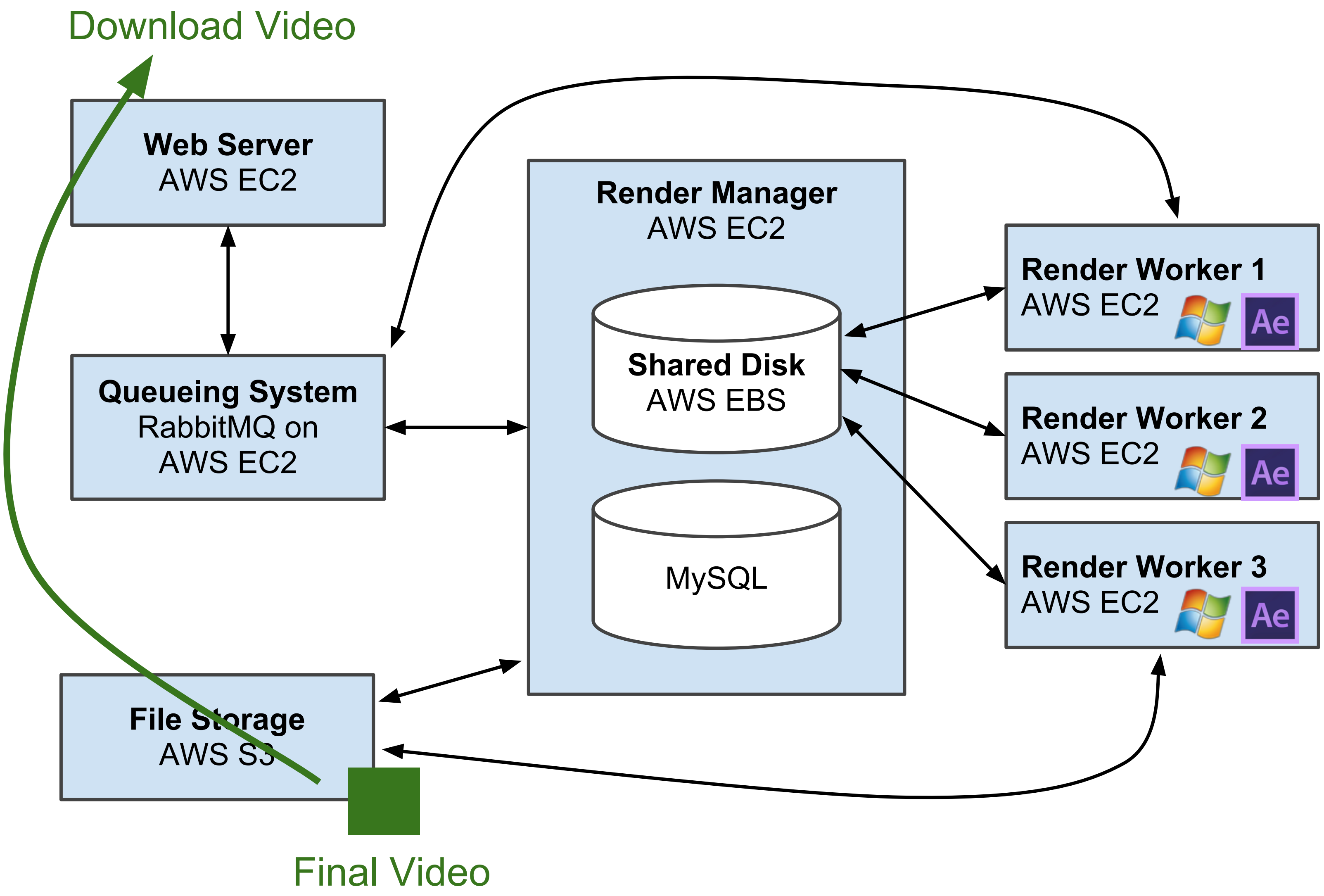
Render Cloud *powered by* amazon web services



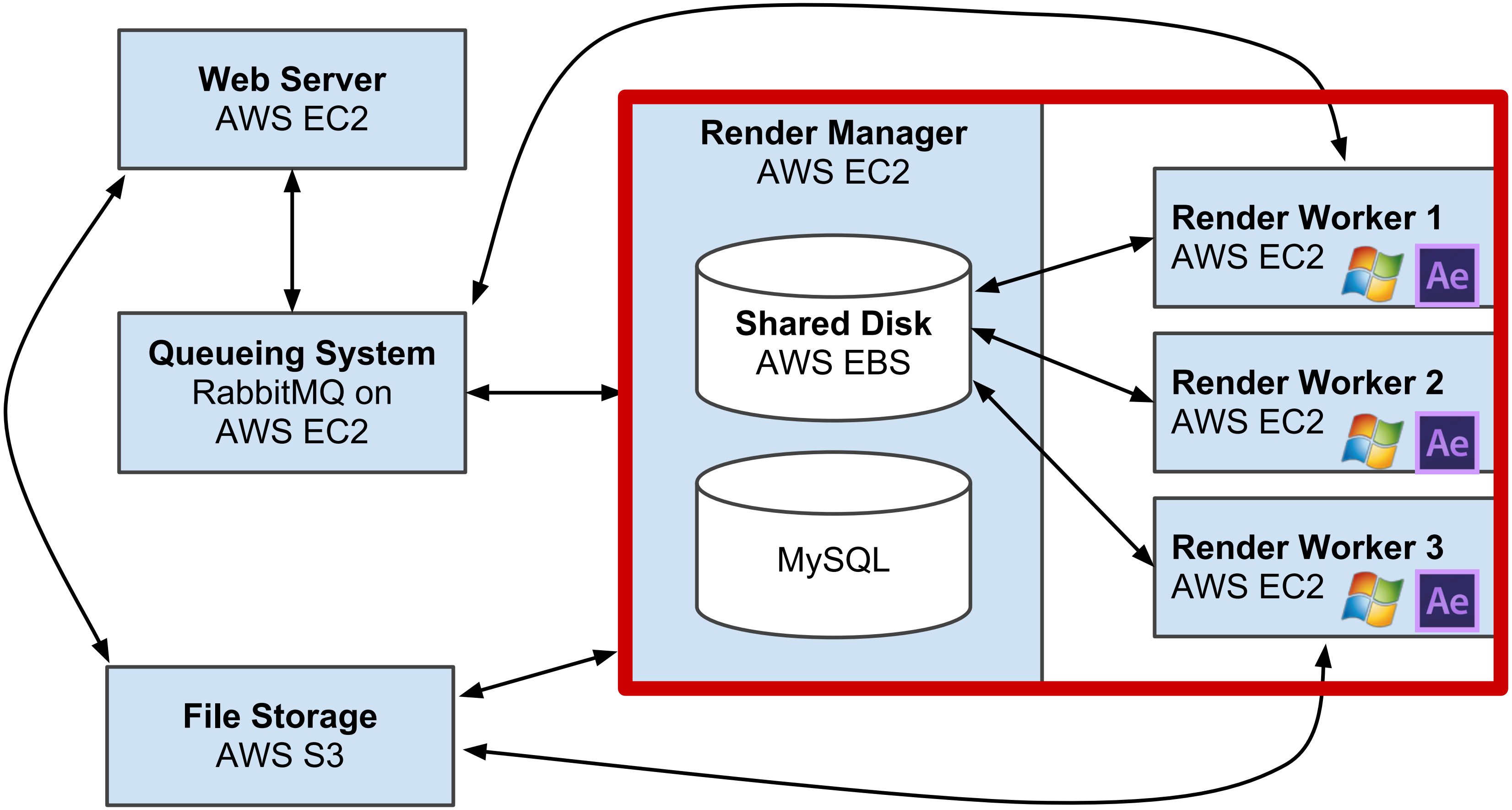
Render Cloud *powered by* amazon web services



Render Cloud *powered by* amazon web services



Queue Consumers



Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

File Storage
AWS S3

Render Manager
AWS EC2

Shared Disk
AWS EBS

MySQL

Render Worker 1
AWS EC2

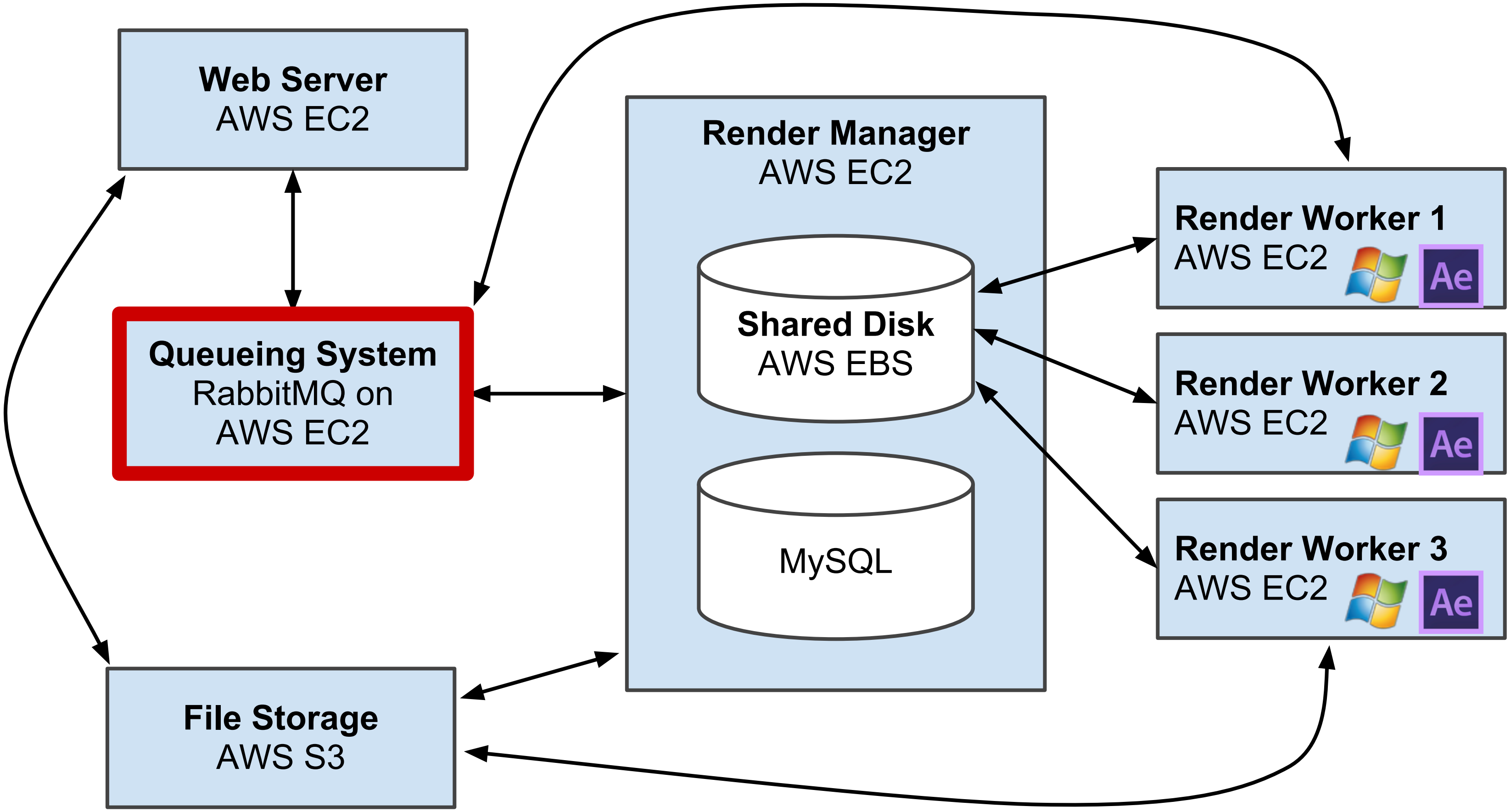
Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

Consumer Thread

```
class ConsumerThread(threading.Thread):

    def __init__(self, queue):
        self.stop = False
        self.queue = queue

    def run(self):
        while not self.stop:
            msg = self.queue.get()
            process(msg)
```

Queueing System



Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

Render Manager
AWS EC2

Shared Disk
AWS EBS

MySQL

Render Worker 1
AWS EC2

Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

File Storage
AWS S3

Queueing System *powered by* RabbitMQ

Messaging that just works

```
import json
import pika

# connect to RabbitMQ
params = pika.URLParameters(
    'amqp://guest:guest@localhost:5672/vhost')
conn = pika.BlockingConnection(params)

channel = conn.channel()

# create a queue named 'WorkerQ'
channel.queue_declare(queue='WorkerQ', durable=True)
```

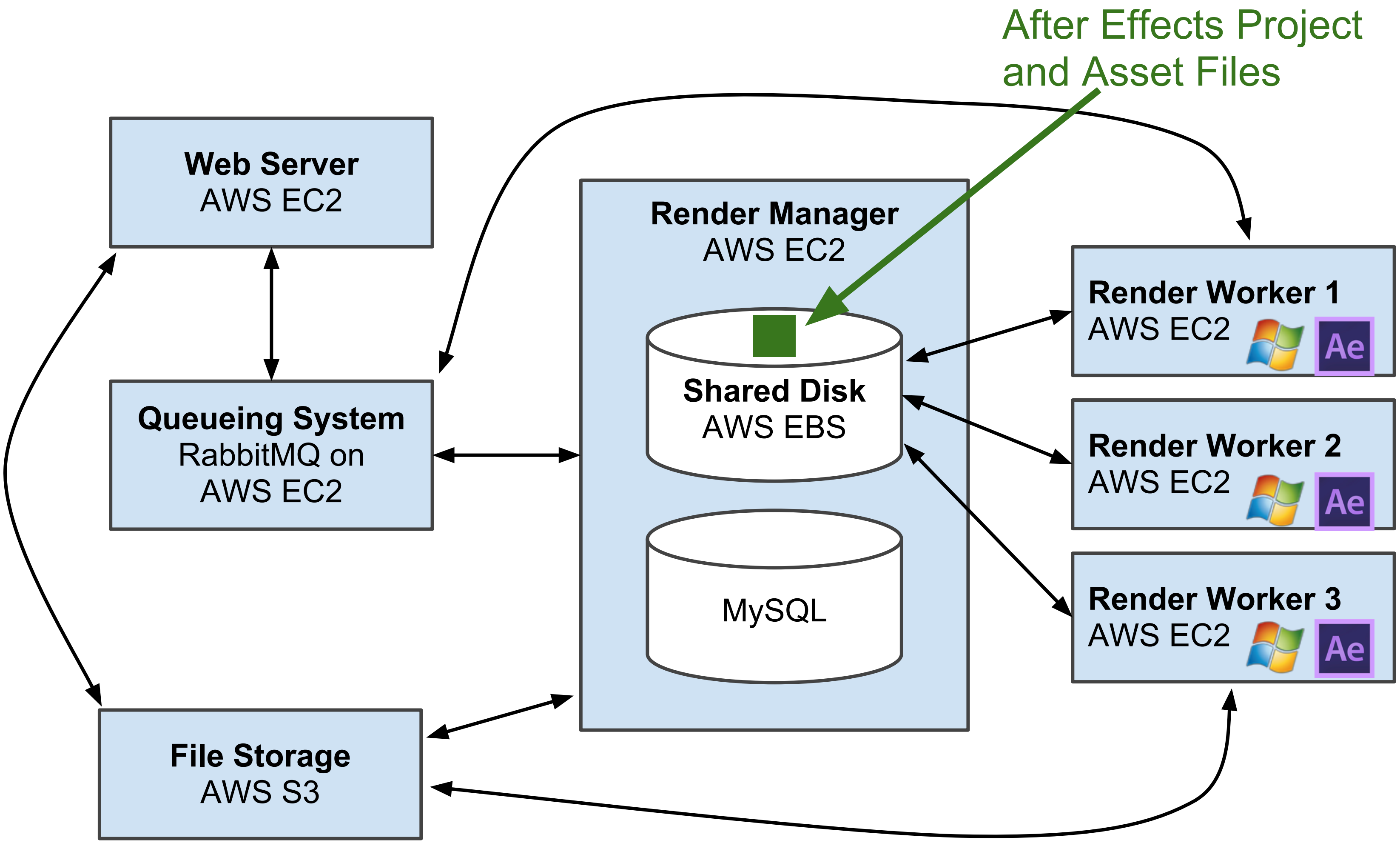
Putting a Message

```
msg = json.dumps({
     'type': 'render',
     'texts': {
         'name': 'Amy Reed',
         'email': 'amy@example.com'
     },
     ...
})

channel.basic_publish(
    exchange='',
    routing_key='WorkerQ',
    body=msg,
    properties=pika.BasicProperties(delivery_mode=2)
)
```

Getting a Message

```
_, _, body = channel.basic_get(queue='WorkerQ')
msg = json.loads(body)

if msg.get('type') == 'render':
     do_render(msg)
```

Preparing Assets

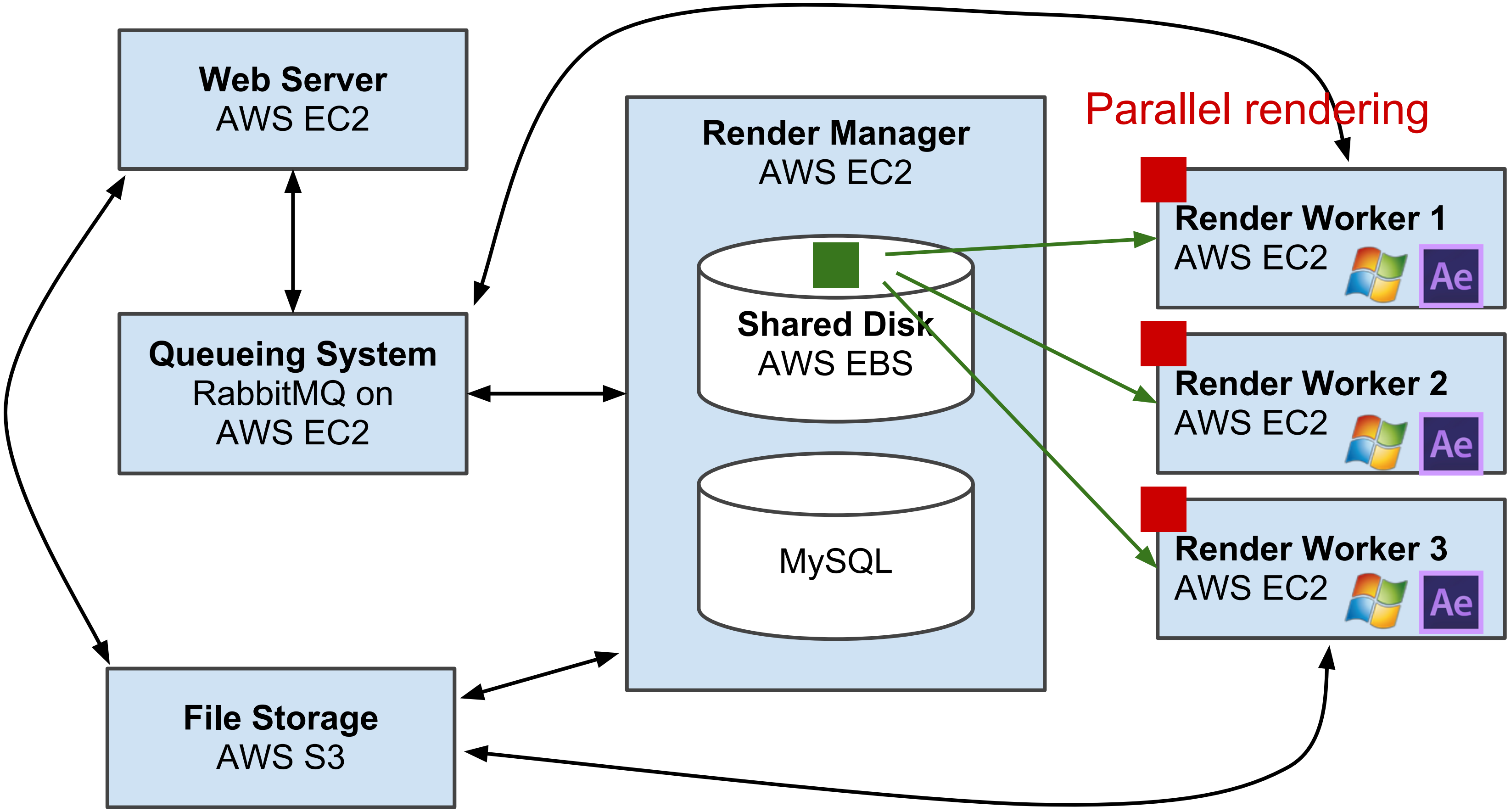


Replacing Text and Photo

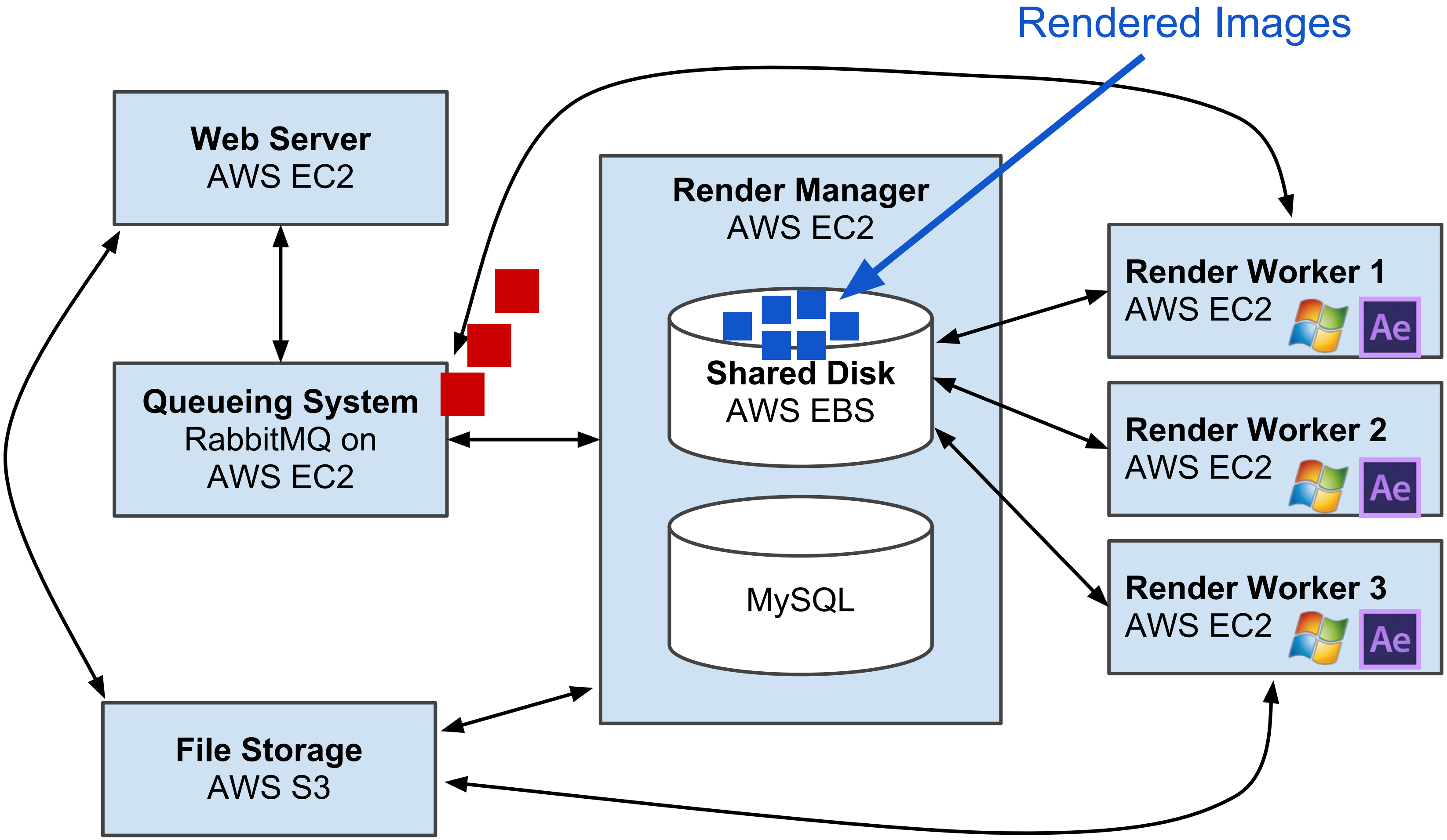
AEPX File (After Effects' XML format)

```
<Layr>
  <string>AMY REED</string>

....

<fileReference fullpath="C:
\path\amy.png" ...>
```

Rendering


Web Server
AWS EC2
Queueing System
RabbitMQ on
AWS EC2
File Storage
AWS S3
Render Manager
AWS EC2
Shared Disk
AWS EBS
MySQL
Parallel rendering
Render Worker 1
AWS EC2
Ae
Render Worker 2
AWS EC2
Ae
Render Worker 3
AWS EC2
Ae

Rendering



Rendered Images

Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

Render Manager
AWS EC2

Shared Disk
AWS EBS

MySQL

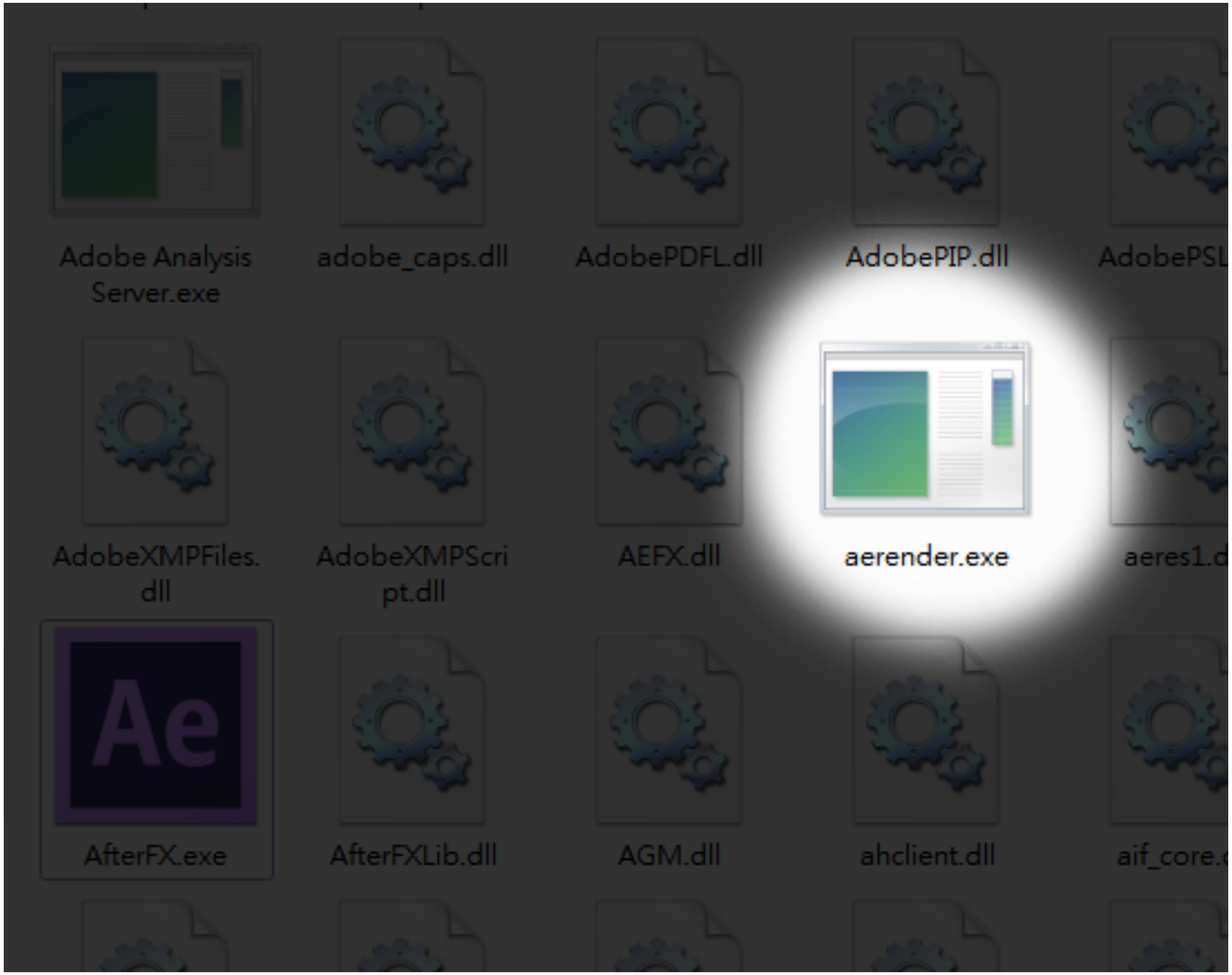
Render Worker 1
AWS EC2

Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

File Storage
AWS S3

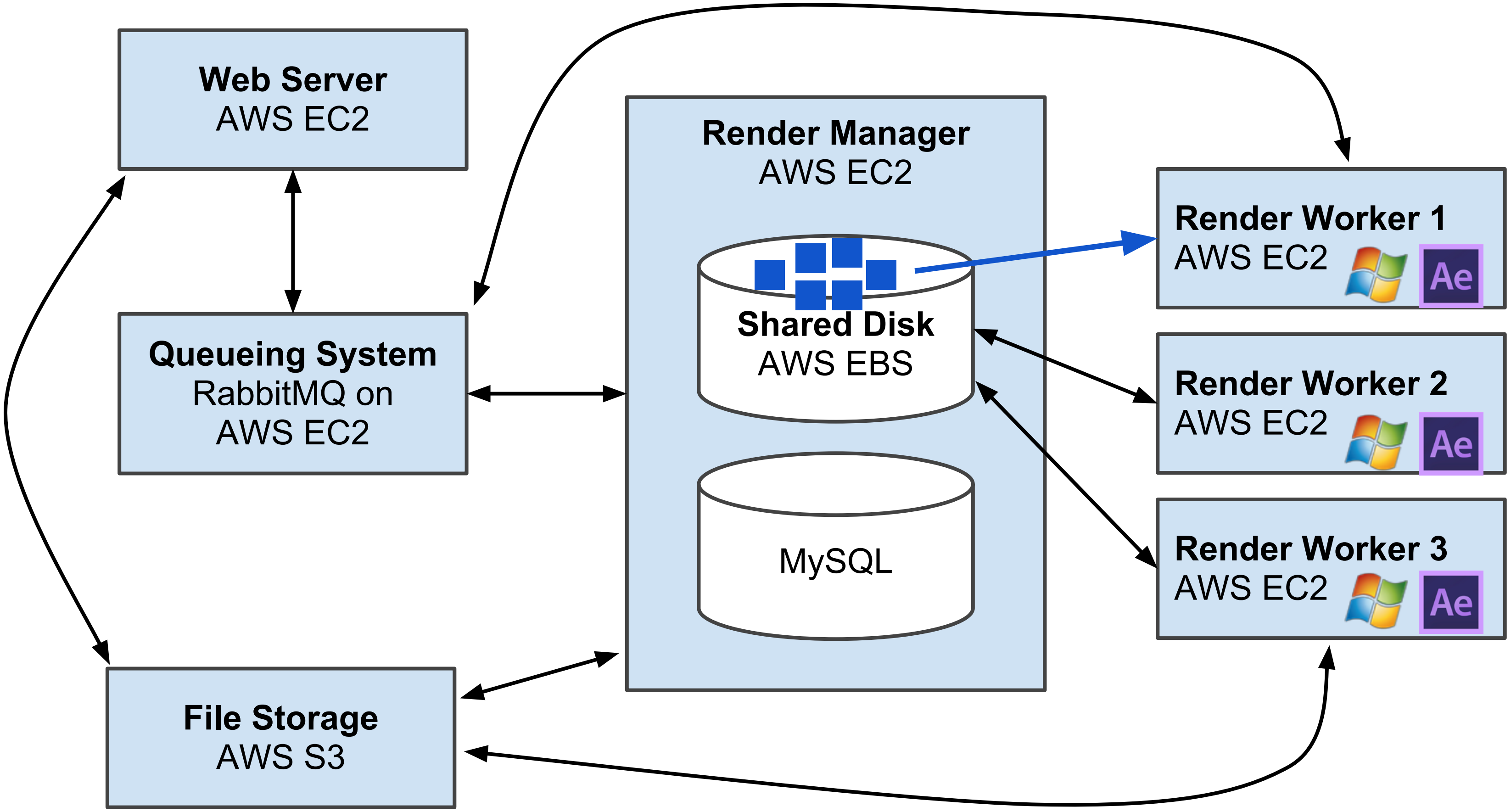
Automated Rendering in After Effects



Calling aerender in Python

```
import subprocess

subprocess.Popen([
    'aerender.exe',
    '-project', 'project.aepx',
    '-output', 'frmae.[#####].jpg',
    '-s', '100',   # start frame
    '-e', '150',   # end frame
    ...
])
```

Merging Images



Web Server
AWS EC2

Queueing System
RabbitMQ on
AWS EC2

File Storage
AWS S3

Render Manager
AWS EC2

Shared Disk
AWS EBS

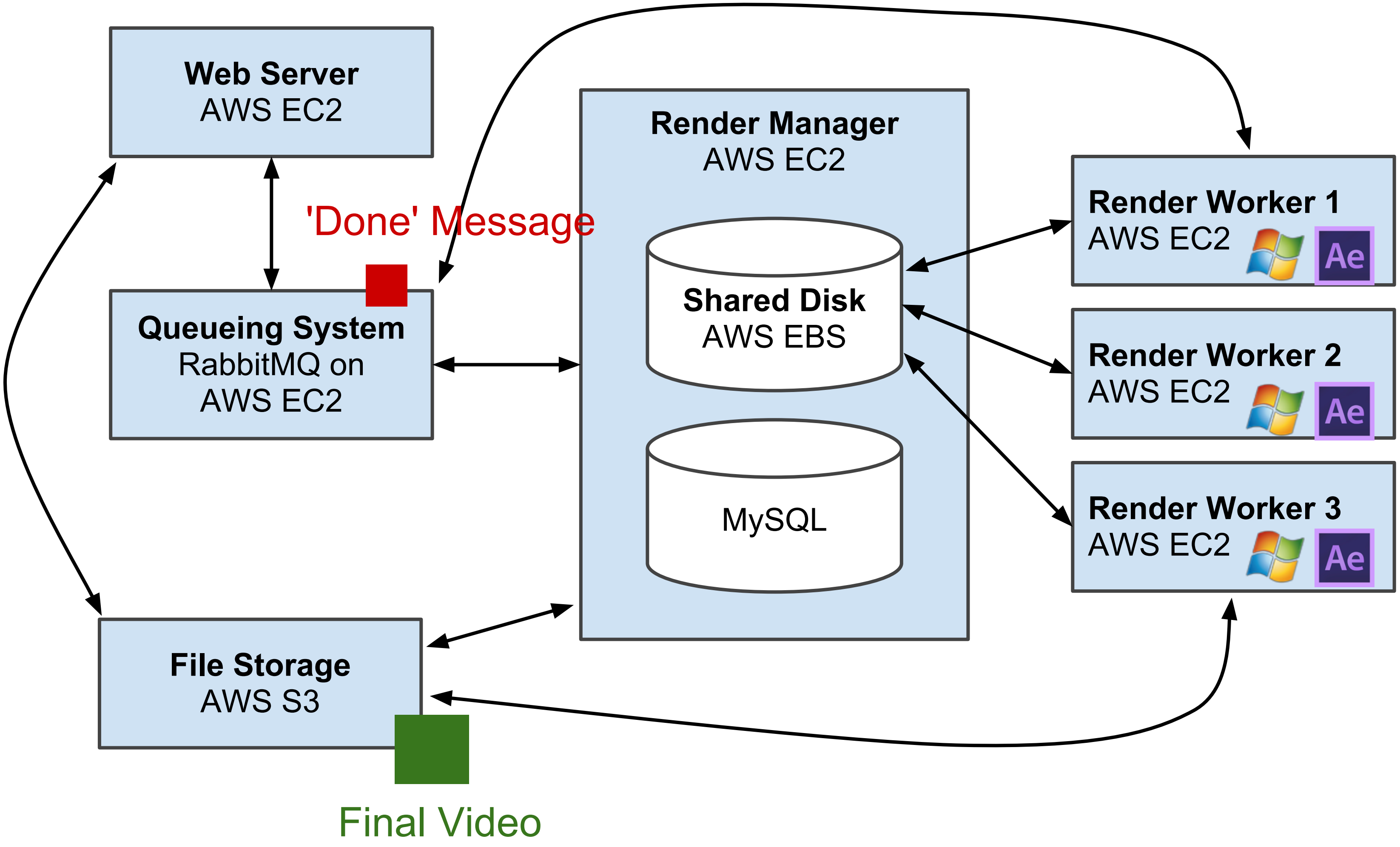
MySQL

Render Worker 1
AWS EC2

Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

Merging Images



Web Server
AWS EC2

'Done' Message

Queueing System
RabbitMQ on
AWS EC2

Render Manager
AWS EC2

Shared Disk
AWS EBS

MySQL

Render Worker 1
AWS EC2

Render Worker 2
AWS EC2

Render Worker 3
AWS EC2

File Storage
AWS S3

Final Video

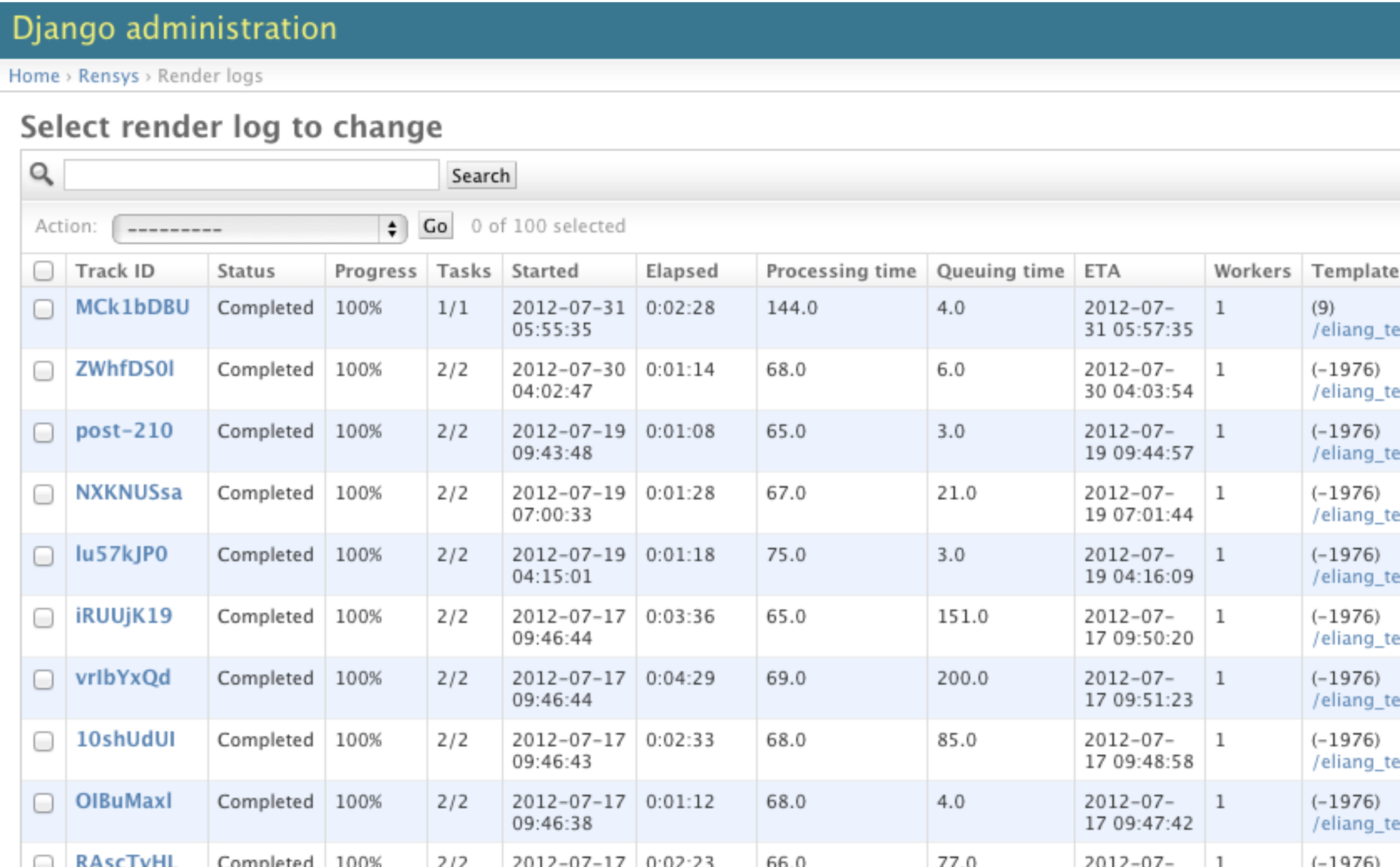
Merging Images

```
import subprocess

subprocess.Popen([
    'ffmpeg',
    '-i', '/path/frame.%05d.jpg',
    '-i', '/path/music.mp3',
    '-s', '720x480',
    '-vcodec', 'libx264',
    '-acodec', 'libaac',
    ...,
    '/path/output.mp4'
])
```

There's More

- Render Watch
- Distributed Locking
- Estimating Render Time

Render Watch



Django administration

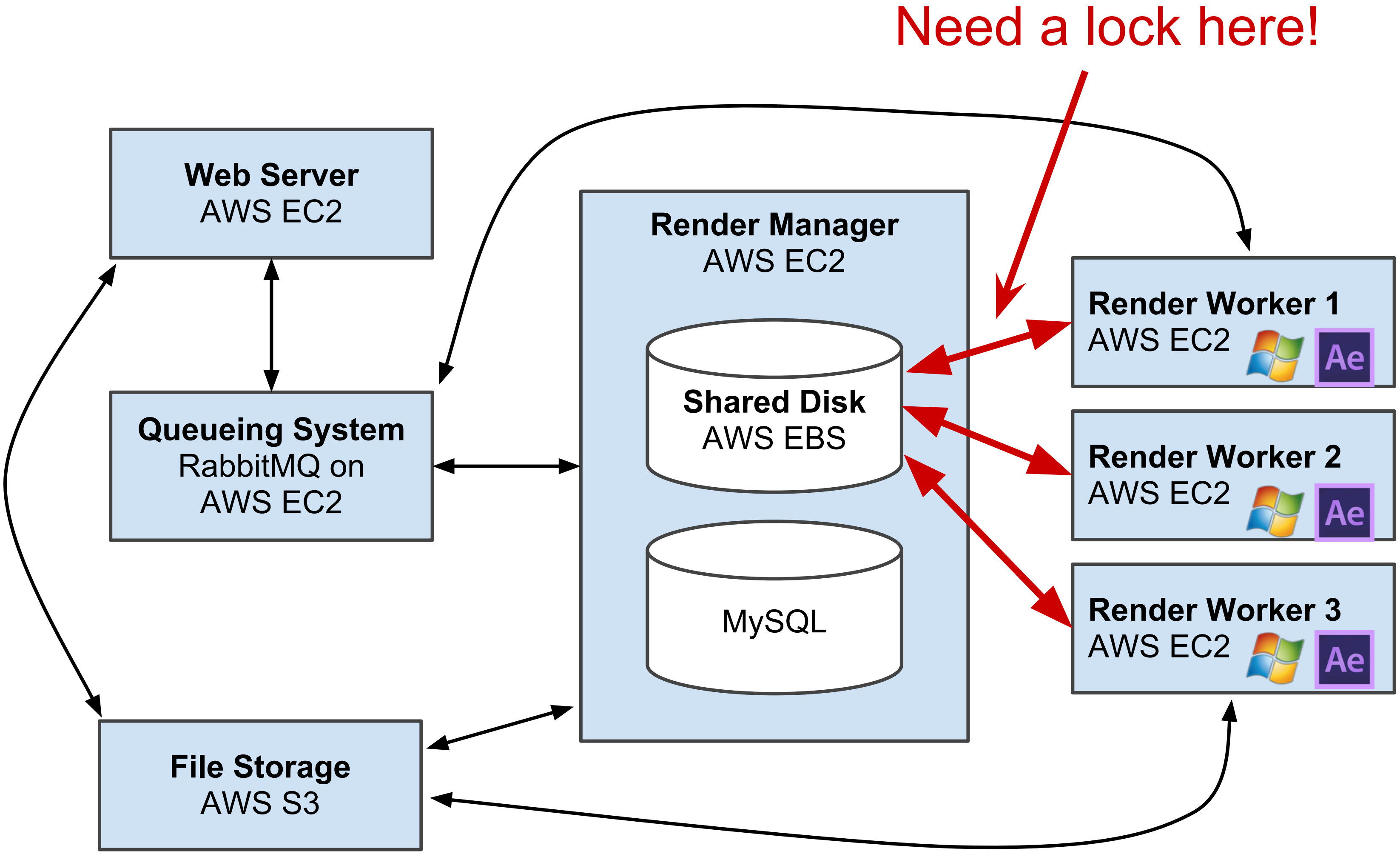
Home › Rensys › Render logs

Select render log to change

0 of 100 selected

☐	Track ID	Status	Progress	Tasks	Started	Elapsed	Processing time	Queuing time	ETA	Workers	Template
☐	MCk1bDBU	Completed	100%	1/1	2012-07-31 05:55:35	0:02:28	144.0	4.0	2012-07-31 05:57:35	1	(9) /eliang_te
☐	ZWhfDS0l	Completed	100%	2/2	2012-07-30 04:02:47	0:01:14	68.0	6.0	2012-07-30 04:03:54	1	(-1976) /eliang_te
☐	post-210	Completed	100%	2/2	2012-07-19 09:43:48	0:01:08	65.0	3.0	2012-07-19 09:44:57	1	(-1976) /eliang_te
☐	NXKNUSsa	Completed	100%	2/2	2012-07-19 07:00:33	0:01:28	67.0	21.0	2012-07-19 07:01:44	1	(-1976) /eliang_te
☐	lu57kJP0	Completed	100%	2/2	2012-07-19 04:15:01	0:01:18	75.0	3.0	2012-07-19 04:16:09	1	(-1976) /eliang_te
☐	iRUUjK19	Completed	100%	2/2	2012-07-17 09:46:44	0:03:36	65.0	151.0	2012-07-17 09:50:20	1	(-1976) /eliang_te
☐	vrIbYxQd	Completed	100%	2/2	2012-07-17 09:46:44	0:04:29	69.0	200.0	2012-07-17 09:51:23	1	(-1976) /eliang_te
☐	10shUdUI	Completed	100%	2/2	2012-07-17 09:46:43	0:02:33	68.0	85.0	2012-07-17 09:48:58	1	(-1976) /eliang_te
☐	OIBuMaxl	Completed	100%	2/2	2012-07-17 09:46:38	0:01:12	68.0	4.0	2012-07-17 09:47:42	1	(-1976) /eliang_te
☐	RAscTyHL	Completed	100%	2/2	2012-07-17	0:02:23	66.0	77.0	2012-07-	1	(-1976)

Distributed Locking



Distributed Locking - Lock

```
def lock(lock_id, lock_timeout):
    lock_path = os.path.join('Z:\\locks', lock_id)
    while True:
        try:
            os.mkdir(lock_path)
        except OSError as err:
            if err.errno not in (errno.EEXIST, errno.EACCES):
                raise err
            try:
                time_locked = os.path.getctime(lock_path)
                if time.time() - time_locked < lock_timeout:
                    time.sleep(1)  # wait for it to be unlocked
                else:
                    os.rmdir(lock_path)  # lock expired
            except OSError:
                pass
```

Distributed Locking - Unlock

```
def unlock(lock_id):
    lock_path = os.path.join('Z:\\locks', lock_id)
    try:
        os.rmdir(lock_path)
    except OSError as err:
        pass
```

Estimating Render Time

num_workers	render_time
2	100
6	x
8	50

Predict the future by reviewing the past:

$$\frac{6 - 2}{8 - 2} = \frac{x - 100}{50 - 100}$$

$$\Rightarrow x = 66.7$$

Summary

- After Effects render cloud on AWS
- Worker queue
- Scalable and parallel
- Python and Django

Thanks!
Q & A