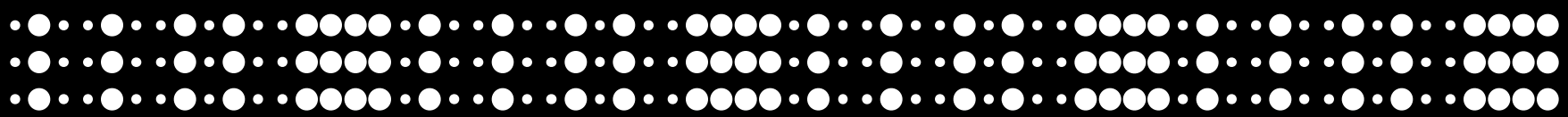

GOOGLE
I/O

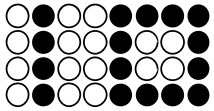


Building Production Quality Apps on App Engine

Ken Ashcraft
5/29/2008

Outline

- Writing code for production
- Testing performance
- Safe deployment after launch



Writing Code for Production

Writing Code for Production

Errors happen

```
movie = Movie()
movie.title = self.request.get('title', None)
movie.director = self.request.get('director', None)
movie.put()
```

Writing Code for Production

Errors happen

```
movie = Movie()
movie.title = self.request.get('title', None)
movie.director = self.request.get('director', None)
movie.put()
```

- Out-of-memory

Writing Code for Production

Errors happen

```
movie = Movie()
movie.title = self.request.get('title', None)
movie.director = self.request.get('director', None)
movie.put()
```

- Out-of-memory
- DeadlineExceeded

Writing Code for Production

Errors happen

```
movie = Movie()
movie.title = self.request.get('title', None)
movie.director = self.request.get('director', None)
movie.put()
```

- Out-of-memory
- DeadlineExceeded
- OverQuotaError

Writing Code for Production

Errors happen

```
movie = Movie()
movie.title = self.request.get('title', None)
movie.director = self.request.get('director', None)
movie.put()
```

- Out-of-memory
- DeadlineExceeded
- OverQuotaError
- Server crash

Writing Code for Production

Errors happen

```
movie = Movie()
movie.title = self.request.get('title', None)
movie.director = self.request.get('director', None)
movie.put()
```

- Out-of-memory
- DeadlineExceeded
- OverQuotaError
- Server crash
- Datastore crash

Writing Code for Production

Errors happen

```
movie = Movie()
movie.title = self.request.get('title', None)
movie.director = self.request.get('director', None)
movie.put()
```

- Out-of-memory
- DeadlineExceeded
- OverQuotaError
- Server crash
- Datastore crash
- Identical entity already exists

Writing Code for Production

Use logging for forensics

```
import logging

user = users.get_current_user()
if user:
  q = db.GqlQuery("SELECT * FROM UserPrefs WHERE user = :1",
                  user)
  results = q.fetch(2)
  if len(results) > 1:
    logging.error("Multiple UserPrefs for user %s", user)
  if len(results) == 0:
    logging.debug("Creating UserPrefs for user %s", user)
    userprefs = UserPrefs(user=user)
    userprefs.put()
  else:
    userprefs = results[0]
else:
  logging.debug("Creating dummy UserPrefs for anonymous user")
```

Writing Code for Production

Email upon exception

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
                             subject='Caught Exception',
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
                             subject='Caught Exception',
                             body=lines)
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
                             subject='Caught Exception',
                             body=lines)
    template_values = {}
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
                             subject='Caught Exception',
                             body=lines)
    template_values = {}
    if users.is_current_user_admin():
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
                             subject='Caught Exception',
                             body=lines)
    template_values = {}
    if users.is_current_user_admin():
      template_values['traceback'] = lines
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
                             subject='Caught Exception',
                             body=lines)
    template_values = {}
    if users.is_current_user_admin():
      template_values['traceback'] = lines
    self.response.out.write(template.render('error.html',
```

Writing Code for Production

Email upon exception

```
class BaseRequestHandler(webapp.RequestHandler):
  def handle_exception(self, exception, debug_mode):
    lines = ''.join(traceback.format_exception(*sys.exc_info
()))
    logging.error(lines)
    mail.send_mail_to_admins(sender='myapp-noreply@gmail.com',
                             subject='Caught Exception',
                             body=lines)
    template_values = {}
    if users.is_current_user_admin():
      template_values['traceback'] = lines
    self.response.out.write(template.render('error.html',
                                            template_values))
```

Writing Code for Production

Using the Python profiler

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
  …
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
  …
  logging.info("Profile data:\n%s", profile_data)
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
  …
  logging.info("Profile data:\n%s", profile_data)

def main():
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
  …
  logging.info("Profile data:\n%s", profile_data)

def main():
  if random.randint(0, 100) == 4:
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
  …
  logging.info("Profile data:\n%s", profile_data)

def main():
  if random.randint(0, 100) == 4:
    profile_main()
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
  …
  logging.info("Profile data:\n%s", profile_data)

def main():
  if random.randint(0, 100) == 4:
    profile_main()
  else:
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

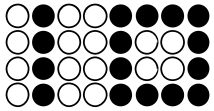
Using the Python profiler

```
def real_main():
  application = webapp.WSGIApplication([
    ('/(.*)', WikiPage),
  ], debug=_DEBUG)
  wsgiref.handlers.CGIHandler().run(application)

def profile_main():
  # Turn on profiling and run real_main()
  …
  logging.info("Profile data:\n%s", profile_data)

def main():
  if random.randint(0, 100) == 4:
    profile_main()
  else:
    real_main()
```

http://code.google.com/appengine/kb/commontasks.html#profiling

Writing Code for Production

- Errors happen!



Loadtests

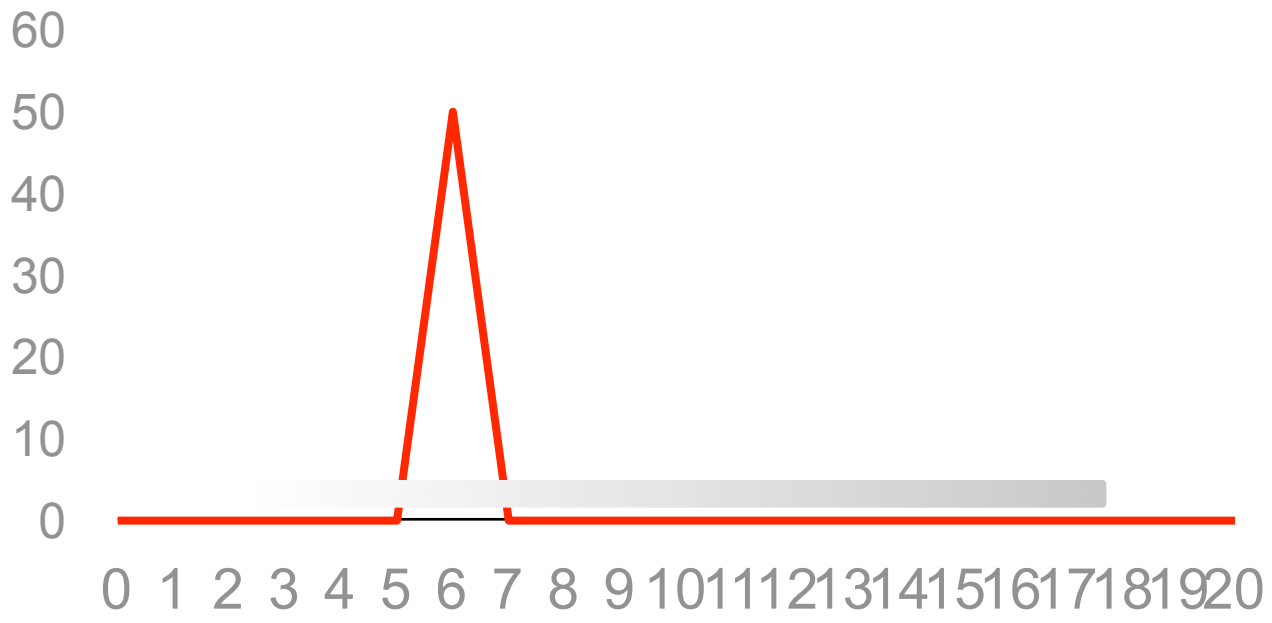
Loadtests

What not to do

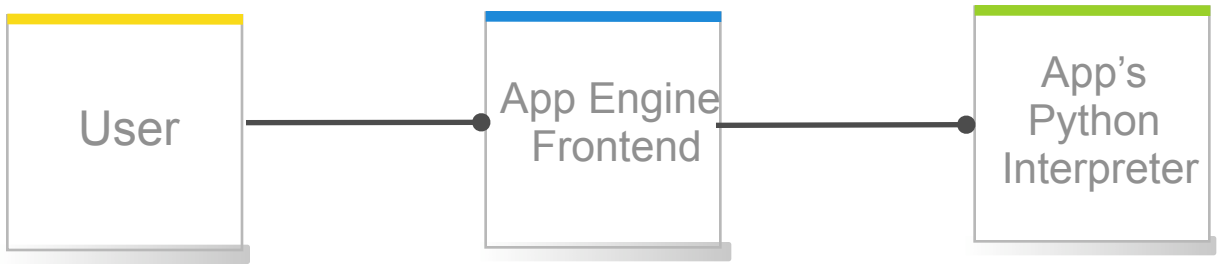
- The dev_appserver is not designed for performance!

Loadtests

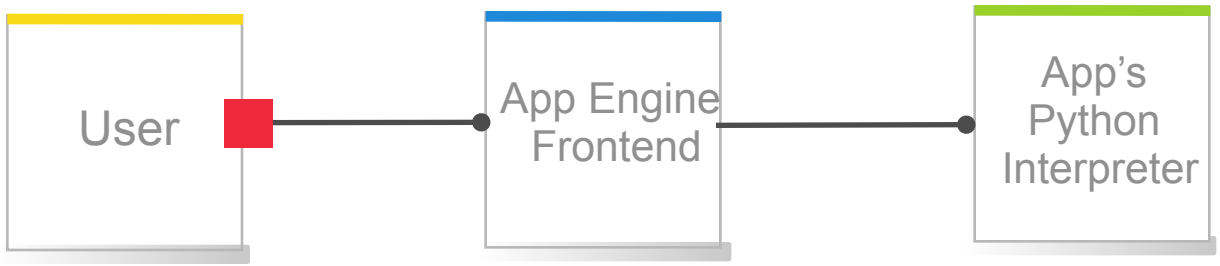
What not to do



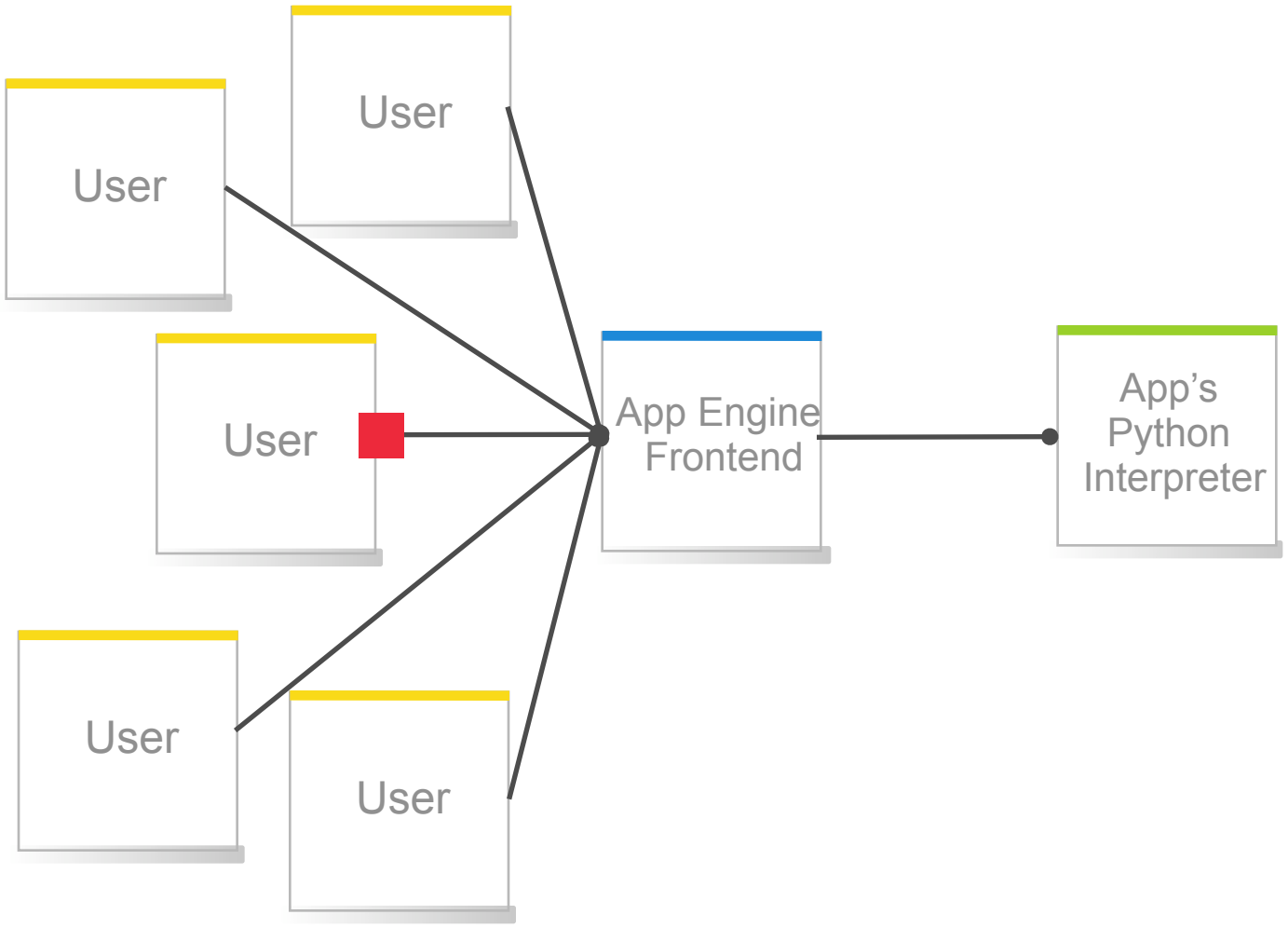
The Request Path



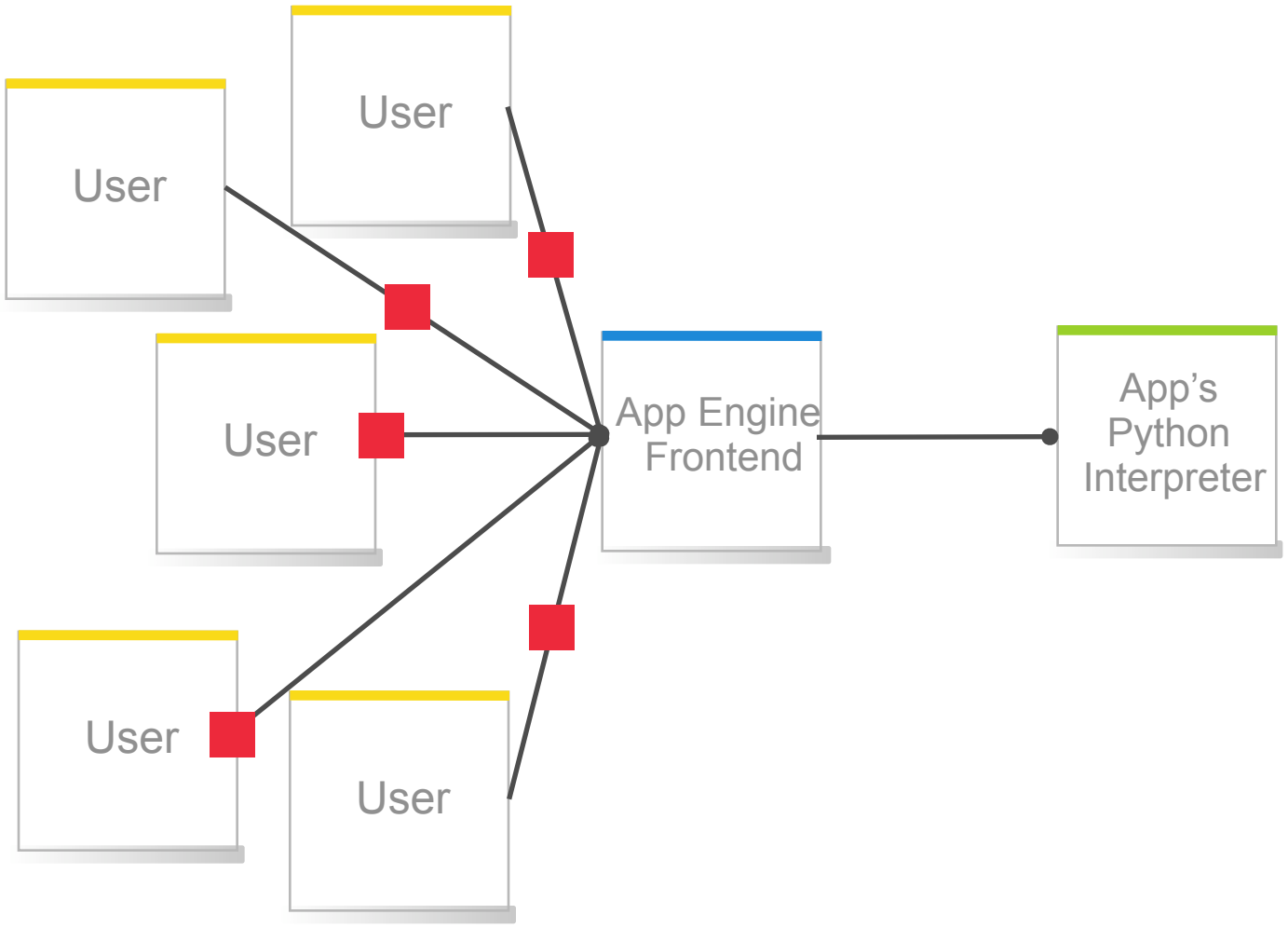
The Request Path



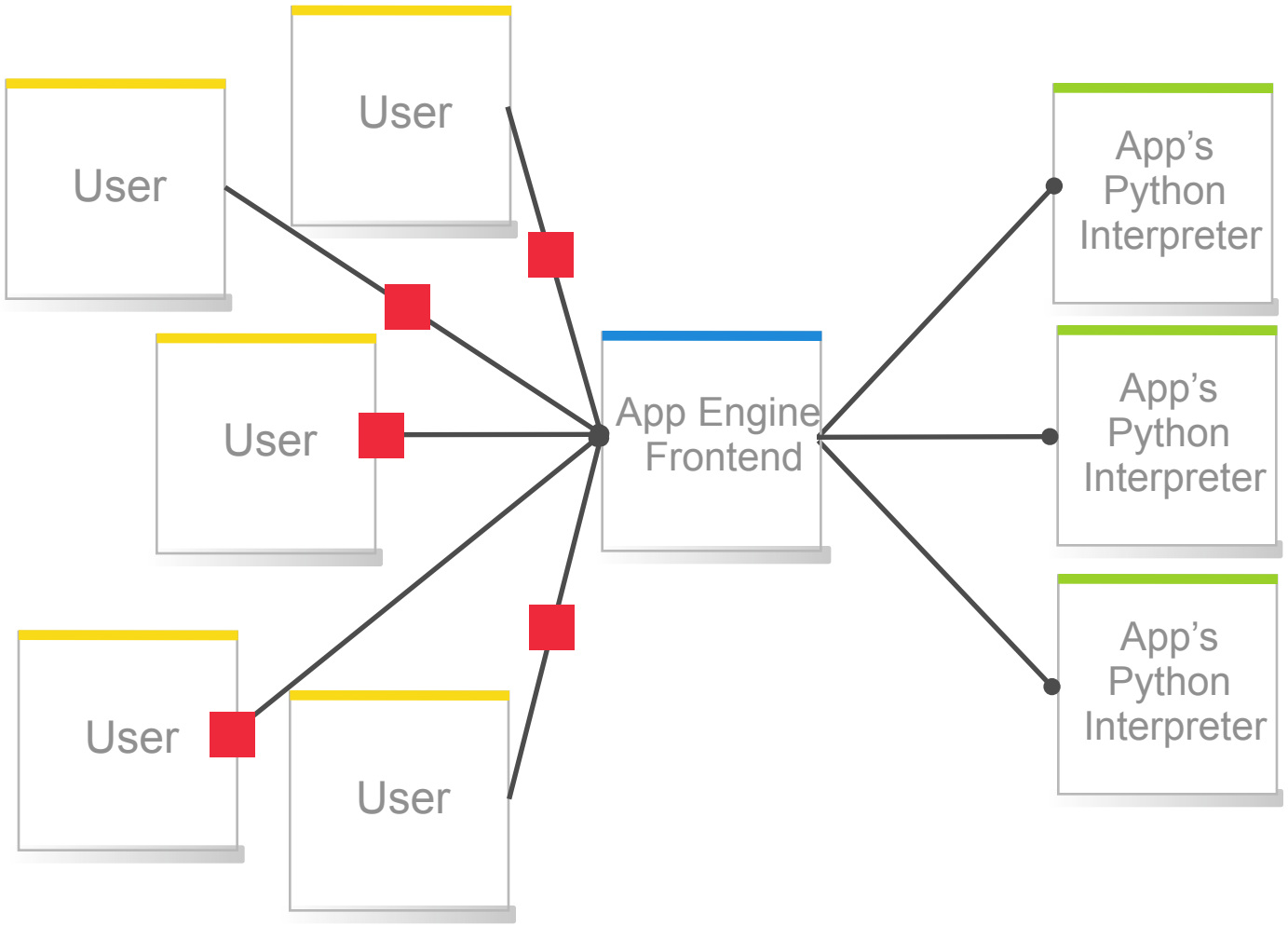
The Request Path



The Request Path

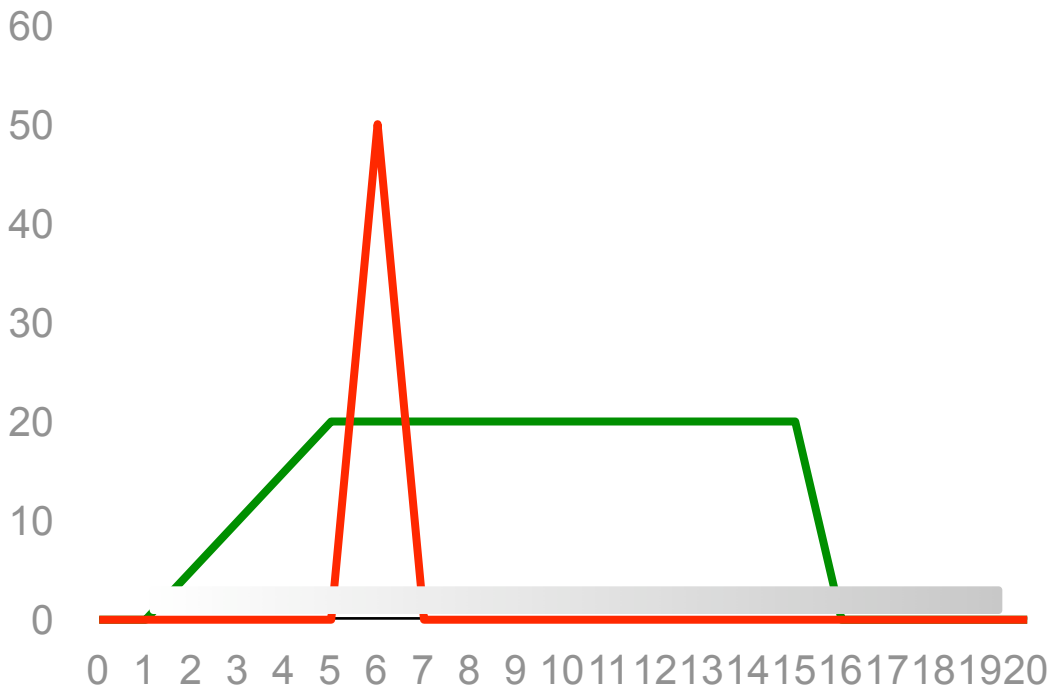


The Request Path



Loadtests

Good vs Bad



Loadtests

Realistic values

Loadtests

Realistic values

- 50,000 users / day

Loadtests

Realistic values

- 50,000 users / day
- 2 pageviews / user

Loadtests

Realistic values

- 50,000 users / day
- 2 pageviews / user
- **100,000 pageviews / day**

Loadtests

Realistic values

- 50,000 users / day
- 2 pageviews / user
- **100,000 pageviews / day**
- 5 requests / pageview

Loadtests

Realistic values

- 50,000 users / day
- 2 pageviews / user
- **100,000 pageviews / day**
- 5 requests / pageview
- 500,000 requests / day

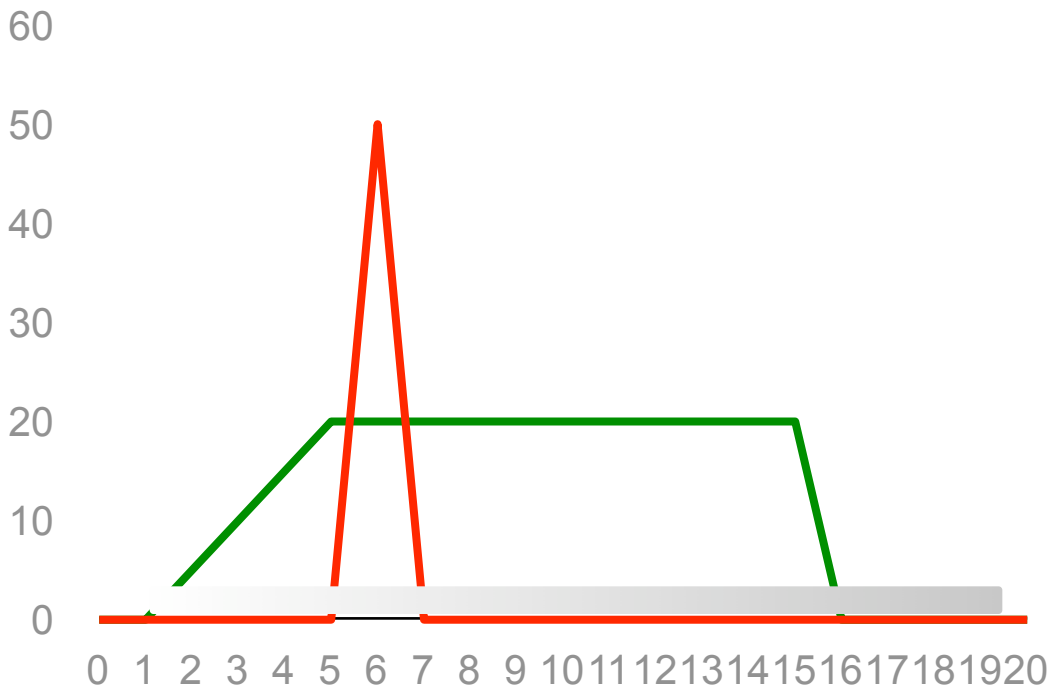
Loadtests

Realistic values

- 50,000 users / day
- 2 pageviews / user
- **100,000 pageviews / day**
- 5 requests / pageview
- 500,000 requests / day
- **5.8 requests / second**

Loadtests

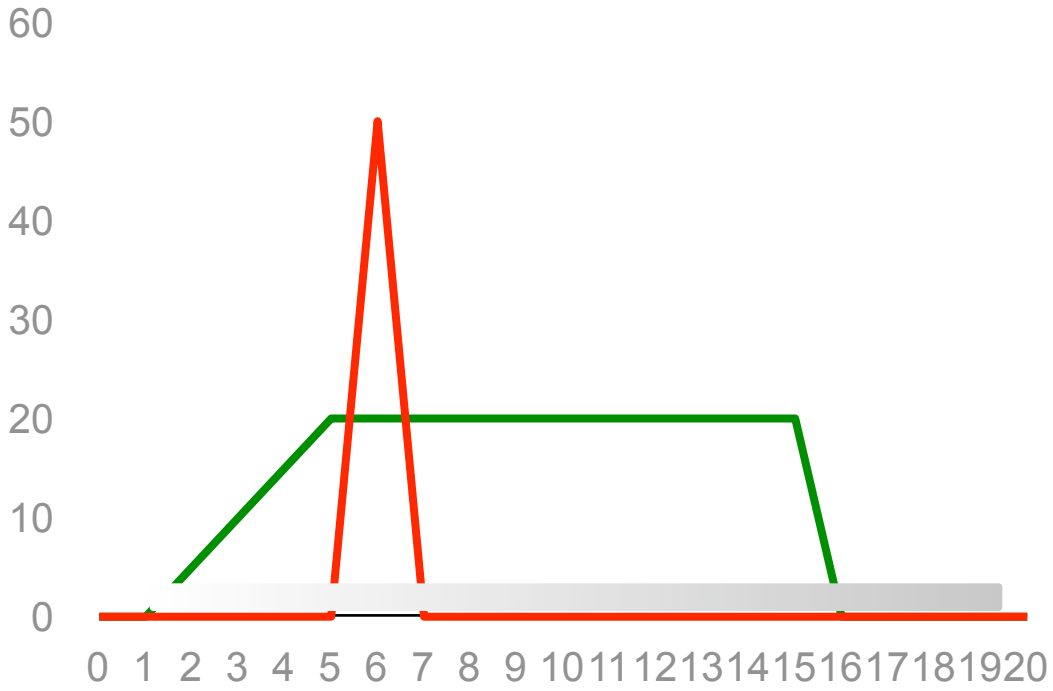
Good vs Bad



Loadtests

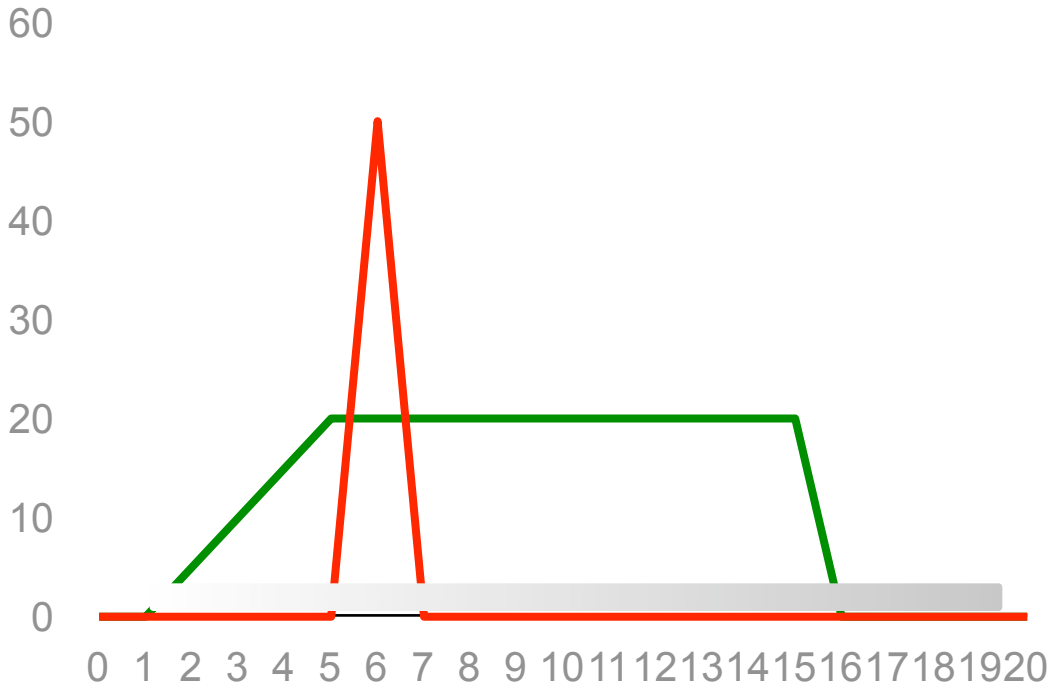
Good vs Bad

- Qualities of a good load test:



Loadtests

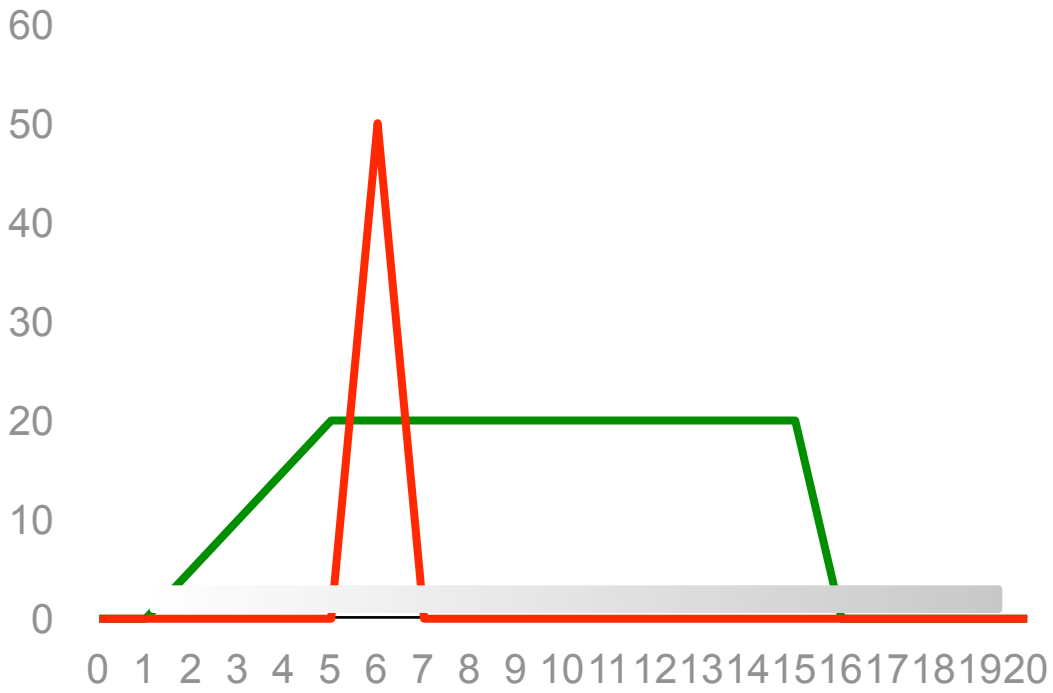
Good vs Bad



- Qualities of a good load test:
  - Use production system (not dev_appserver)

Loadtests

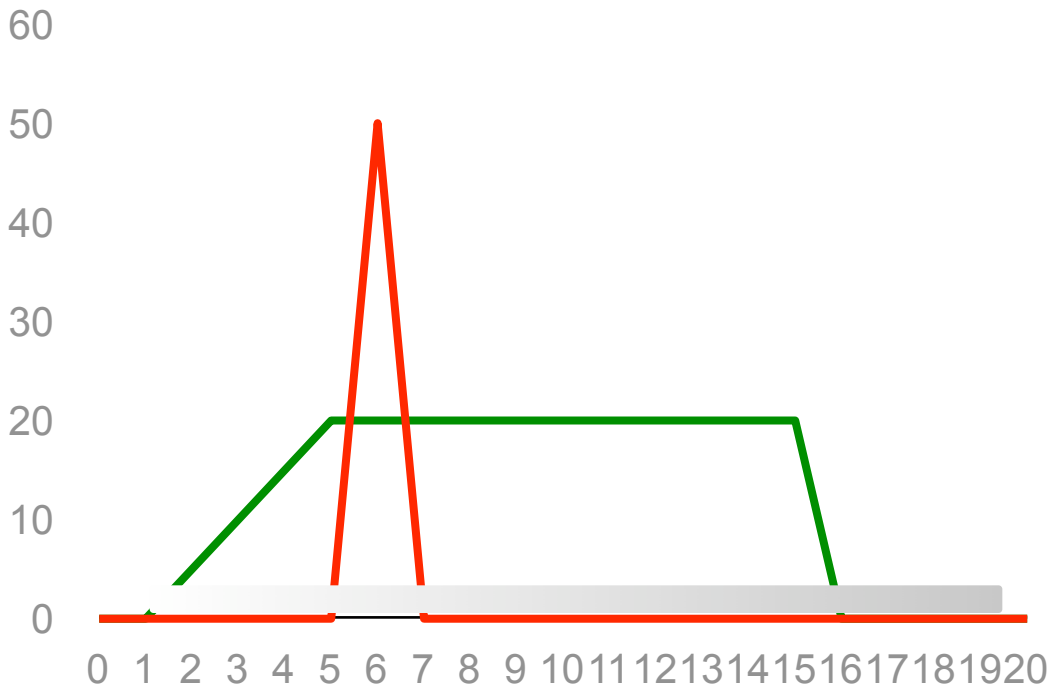
Good vs Bad



- Qualities of a good load test:
  - Use production system (not dev_appserver)
  - Gradual ramp up

Loadtests

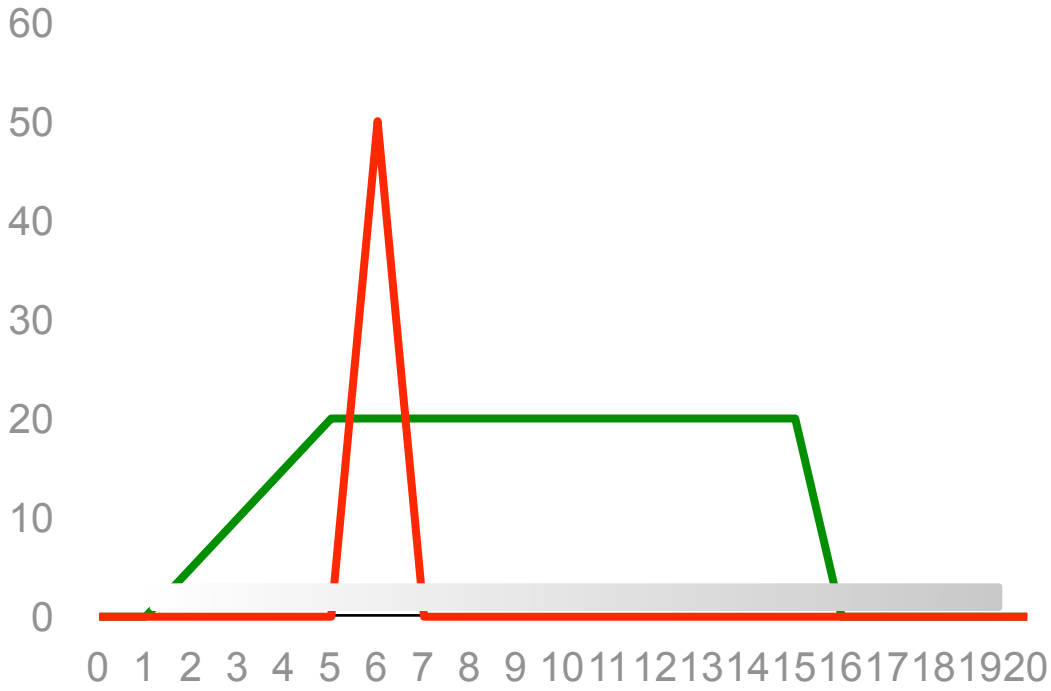
Good vs Bad



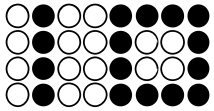
- Qualities of a good load test:
  - Use production system (not dev_appserver)
  - Gradual ramp up
  - Sustained load

Loadtests

Good vs Bad



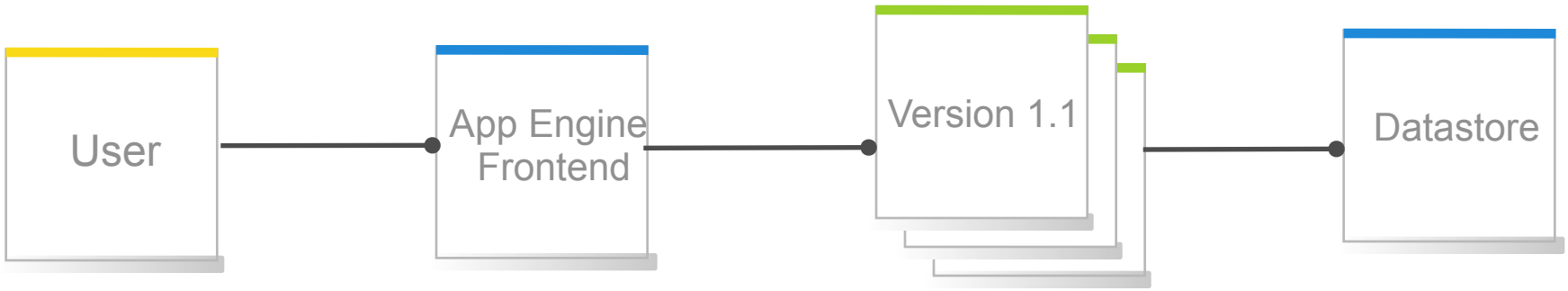
- Qualities of a good load test:
  - Use production system (not dev_appserver)
  - Gradual ramp up
  - Sustained load
  - Realistic load



Safe Deployment

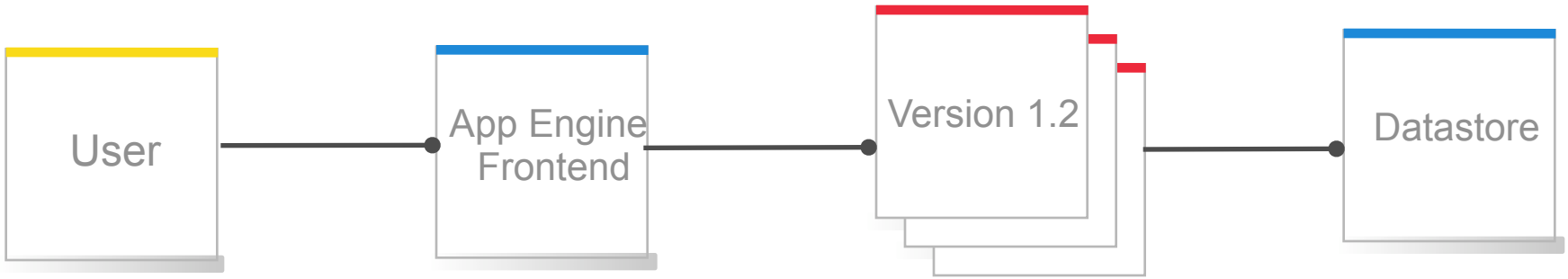
How App Versioning Works

http://wiki-io.appspot.com

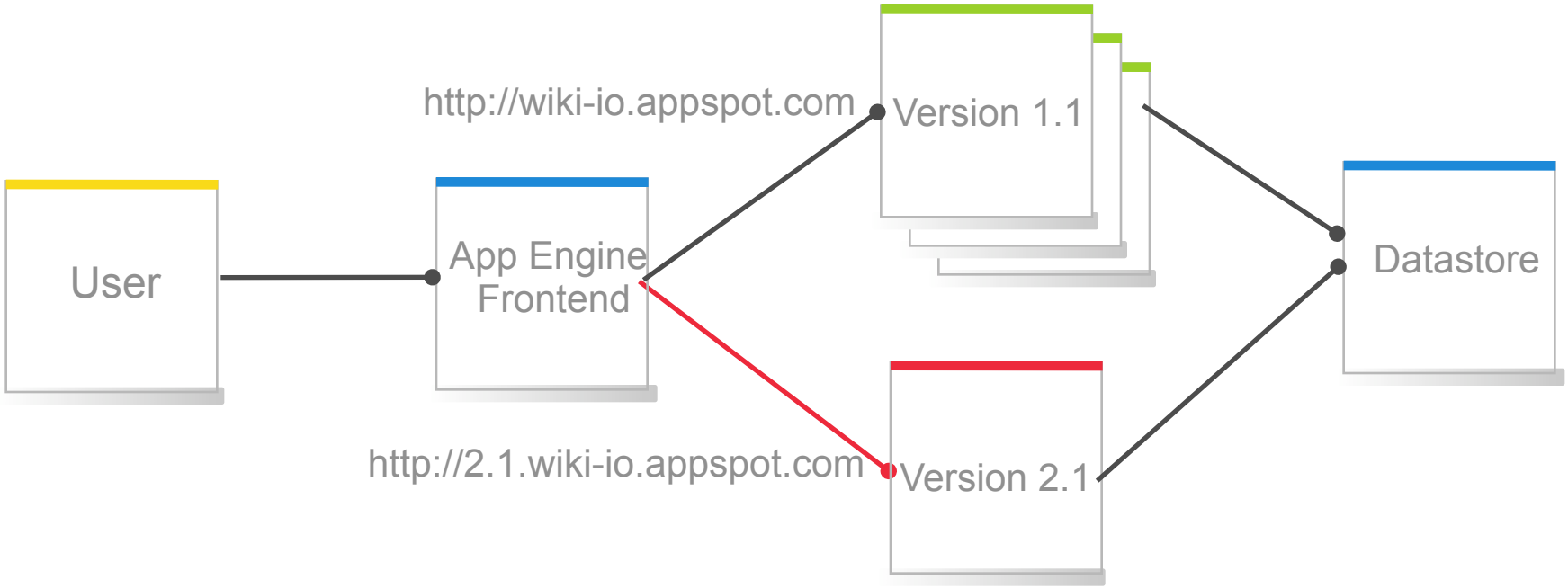


How App Versioning Works

http://wiki-io.appspot.com



How App Versioning Works



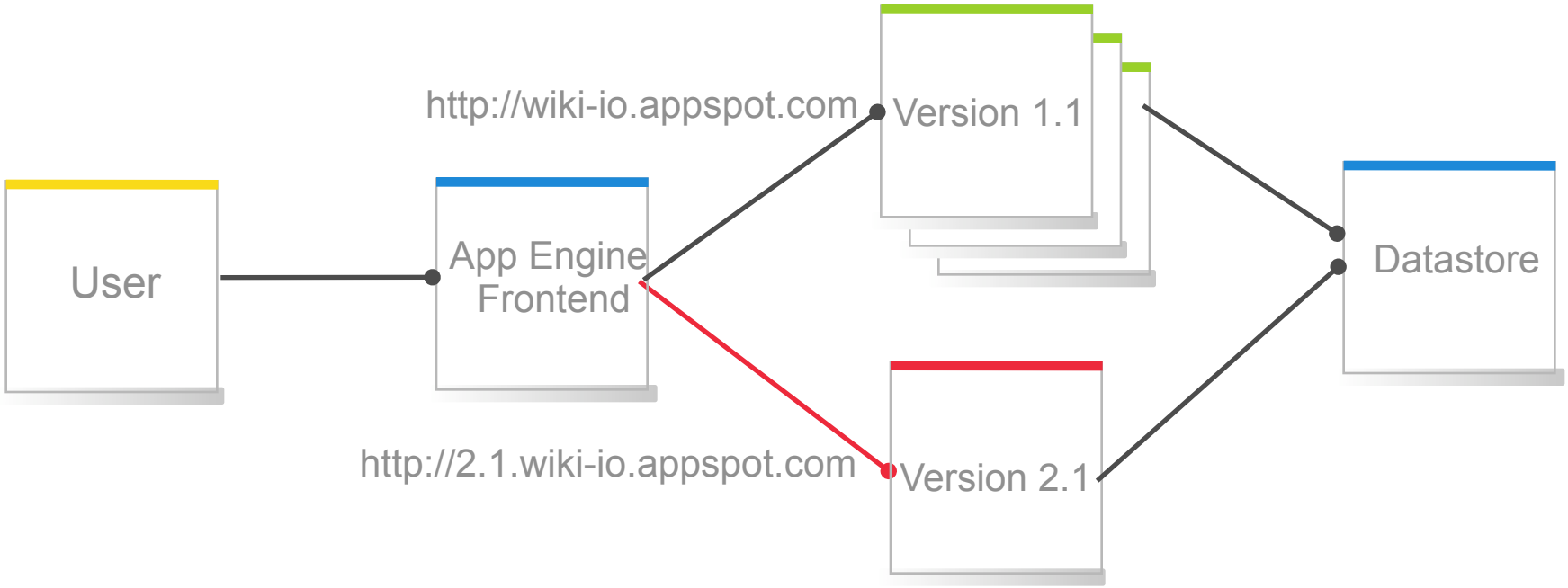
App Versioning

Control the version with app.yaml

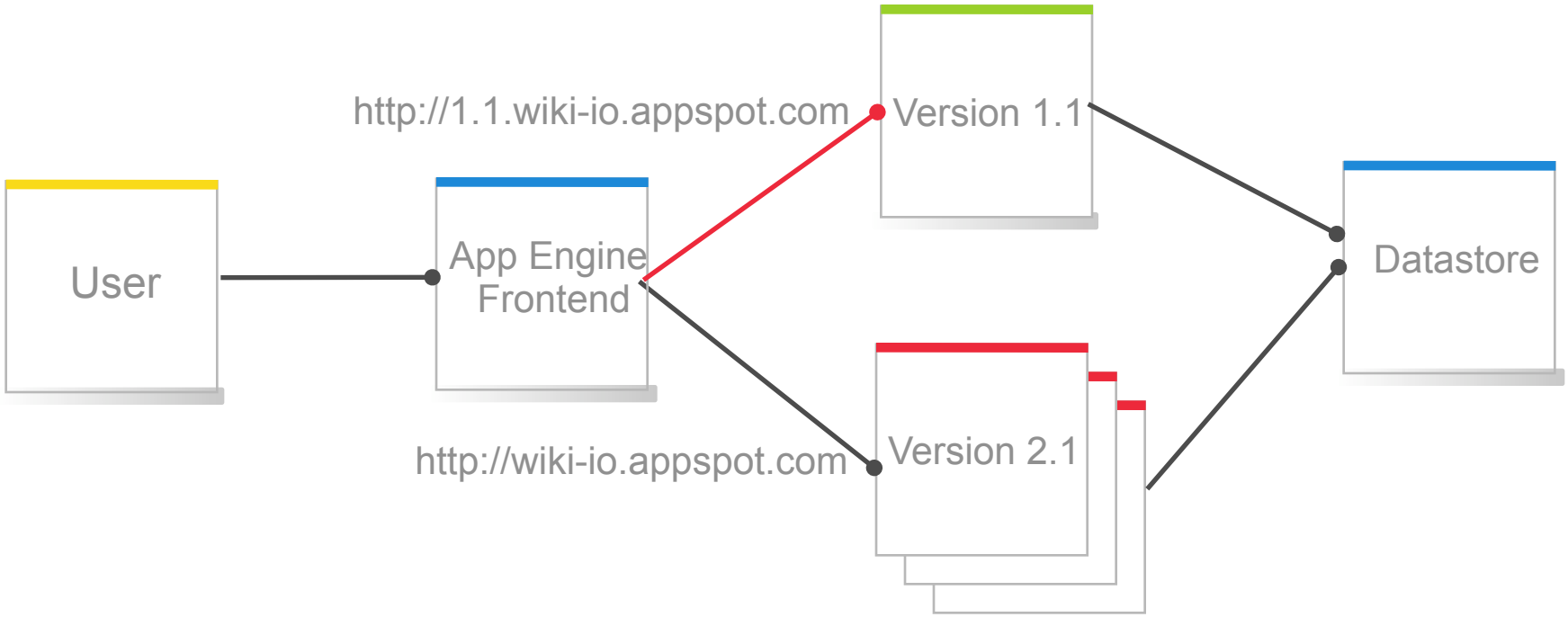
```
application: wiki-io
version: 5482  # Revision number
runtime: python
api_version: 1
…
```

- Suggestion: Create a build script to populate *version*

How App Versioning Works



How App Versioning Works



Testing the Release

- http://<major>.<minor>.<app>.appspot.com
- Selenium: a test tool for web applications
  - http://selenium.openqa.org
- Use the log viewer to look for errors

Feature Requests

Feature Requests

- A/B testing

Feature Requests

- A/B testing
- Integration with SVN

Feature Requests

- A/B testing
- Integration with SVN
- Edit running code

Summary

Summary

- Write code to handle errors
  - Use logging and catch exceptions

Summary

- Write code to handle errors
  - Use logging and catch exceptions
- Do realistic loadtests

Summary

- Write code to handle errors
  - Use logging and catch exceptions
- Do realistic loadtests
- Use versions for safe deployment


GOOGLE
I/O