Q. Use Google App Engine.

app.yaml

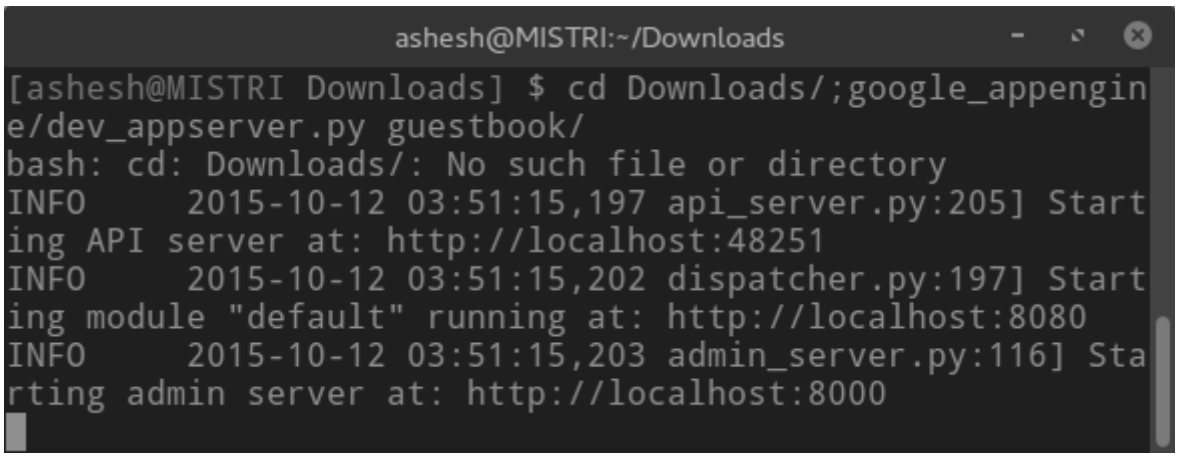
```
version: 1
runtime: python27
api_version: 1
threadsafe: true

handlers:
- url: /.*
  script: guestbook.app

libraries:
- name: webapp2
  version: latest
```

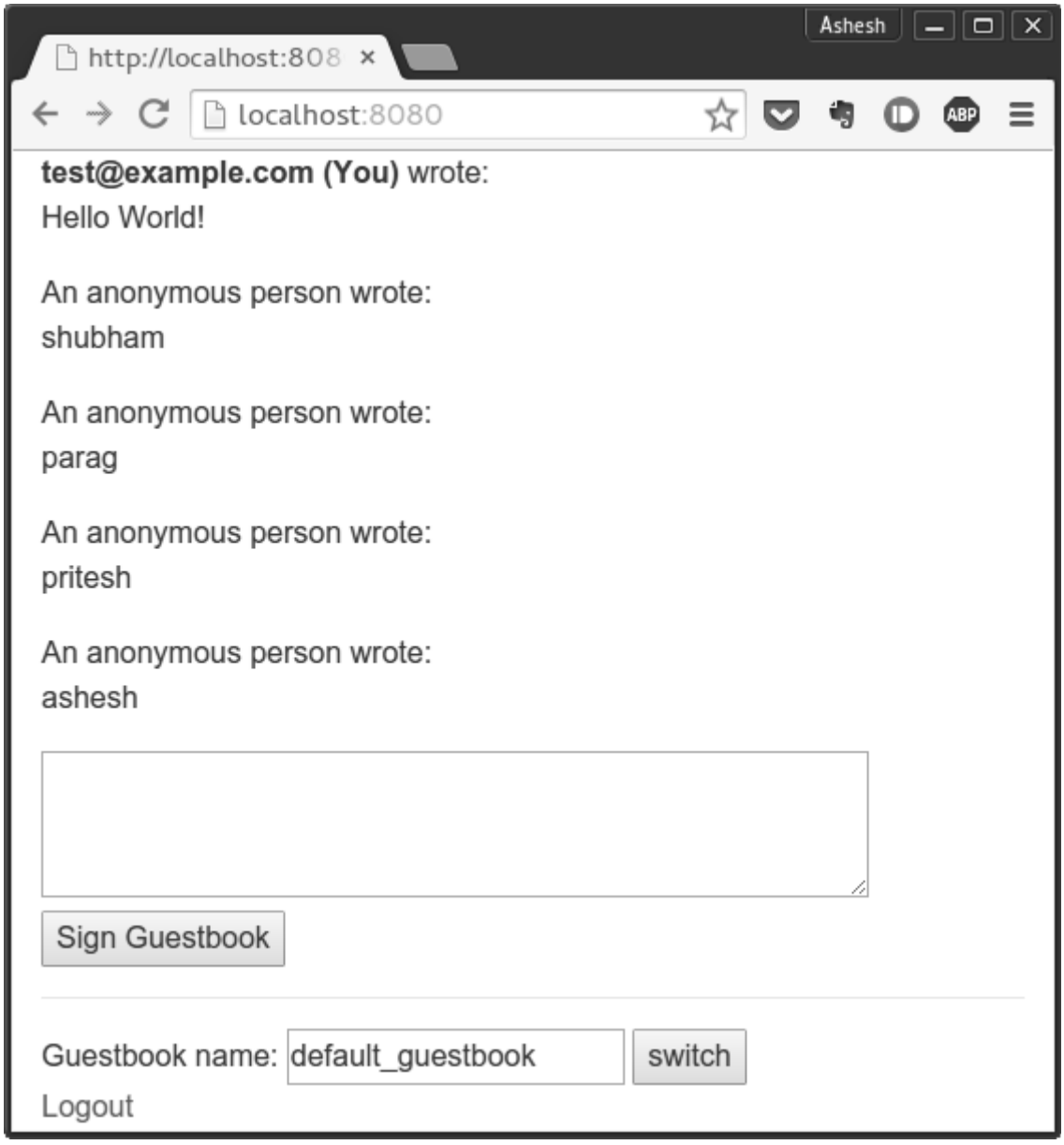
guestbook.py

```
import cgi
import urllib

from google.appengine.api import users
from google.appengine.ext import ndb

import webapp2

MAIN_PAGE_FOOTER_TEMPLATE = """\
    <form action="/sign?%s" method="post">
      <div><textarea name="content" rows="3" cols="60"></textarea></div>
      <div><input type="submit" value="Sign Guestbook"></div>
    </form>
    <hr>
    <form>Guestbook name:
      <input value="%s" name="guestbook_name">
      <input type="submit" value="switch">
    </form>
    <a href="%s">%s</a>
  </div>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
alpha/js/bootstrap.min.js" integrity="sha256-
+h0g0j7qusP72OZaLPCSZ5wjZLnoUUicoxbvrl14WxM= sha512-
0z9zJIjxQaDVzlysxlaqkZ8L9jh8jZ2d54F3Dn36Y0a8C6eI/RFOME/tLCFJ42hfOxdclfa29
lPSNCmX5ekxnw==" crossorigin="anonymous"></script>
  </body>
</html>
"""

DEFAULT_GUESTBOOK_NAME = 'default_guestbook'

# We set a parent key on the 'Greetings' to ensure that they are all
# in the same entity group. Queries across the single entity group
# will be consistent.  However, the write rate should be limited to
```

```
# ~1/second.

def guestbook_key(guestbook_name=DEFAULT_GUESTBOOK_NAME):
    """Constructs a Datastore key for a Guestbook entity.

    We use guestbook_name as the key.
    """
    return ndb.Key('Guestbook', guestbook_name)


class Author(ndb.Model):
    """Sub model for representing an author."""
    identity = ndb.StringProperty(indexed=False)
    email = ndb.StringProperty(indexed=False)


class Greeting(ndb.Model):
    """A main model for representing an individual Guestbook entry."""
    author = ndb.StructuredProperty(Author)
    content = ndb.StringProperty(indexed=False)
    date = ndb.DateTimeProperty(auto_now_add=True)


class MainPage(webapp2.RequestHandler):
    def get(self):
        self.response.write('<!DOCTYPE html><html><link
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
alpha/css/bootstrap.min.css" rel="stylesheet" integrity="sha256-
GHW2S7IZAQe8+YkyL99LyDj1zdWXSPOG+JZafCtKiSc= sha512-
vxM32w6T7zJ83xOQ6FT4CEFnlasqmkcB0+ojgbI0N6ZtSxYvHmT7sX2icN07TqEqr5wdKwoLk
mB8sAsGAjCJHg==" crossorigin="anonymous"><div class="container-
fluid"><body>')
        guestbook_name = self.request.get('guestbook_name',
                                          DEFAULT_GUESTBOOK_NAME)

        # Ancestor Queries, as shown here, are strongly consistent
        # with the High Replication Datastore. Queries that span
        # entity groups are eventually consistent. If we omitted the
        # ancestor from this query there would be a slight chance that
        # Greeting that had just been written would not show up in a
        # query.
        greetings_query = Greeting.query(
            ancestor=guestbook_key(guestbook_name)).order(-Greeting.date)
        greetings = greetings_query.fetch(10)

        user = users.get_current_user()
        for greeting in greetings:
            if greeting.author:
                author = greeting.author.email
                if user and user.user_id() == greeting.author.identity:
                    author += ' (You)'
```

```
                self.response.write('<b>%s</b> wrote:' % author)
            else:
                self.response.write('An anonymous person wrote:')
            self.response.write('<blockquote>%s</blockquote>' %
                                cgi.escape(greeting.content))

        if user:
            url = users.create_logout_url(self.request.uri)
            url_linktext = 'Logout'
        else:
            url = users.create_login_url(self.request.uri)
            url_linktext = 'Login'

        # Write the submission form and the footer of the page
        sign_query_params = urllib.urlencode({'guestbook_name':
                                              guestbook_name})
        self.response.write(MAIN_PAGE_FOOTER_TEMPLATE %
                            (sign_query_params,
cgi.escape(guestbook_name),
                             url, url_linktext))

class Guestbook(webapp2.RequestHandler):
    def post(self):
        # We set the same parent key on the 'Greeting' to ensure each
        # Greeting is in the same entity group. Queries across the
        # single entity group will be consistent. However, the write
        # rate to a single entity group should be limited to
        # ~1/second.
        guestbook_name = self.request.get('guestbook_name',
                                          DEFAULT_GUESTBOOK_NAME)
        greeting = Greeting(parent=guestbook_key(guestbook_name))

        if users.get_current_user():
            greeting.author = Author(
                    identity=users.get_current_user().user_id(),
                    email=users.get_current_user().email())

        greeting.content = self.request.get('content')
        greeting.put()

        query_params = {'guestbook_name': guestbook_name}
        self.redirect('/?' + urllib.urlencode(query_params))

app = webapp2.WSGIApplication([
    ('/', MainPage),
    ('/sign', Guestbook),
], debug=True)
```


ashesh@MISTRI:~/Downloads
[ashesh@MISTRI Downloads] $ cd Downloads/;google_appengin
e/dev_appserver.py guestbook/
bash: cd: Downloads/: No such file or directory
INFO 2015-10-12 03:51:15,197 api_server.py:205] Start
ing API server at: http://localhost:48251
INFO 2015-10-12 03:51:15,202 dispatcher.py:197] Start
ing module "default" running at: http://localhost:8080
INFO 2015-10-12 03:51:15,203 admin_server.py:116] Sta
rting admin server at: http://localhost:8000


Ashesh
http://localhost:808
localhost:8080
test@example.com (You) wrote:
Hello World!
An anonymous person wrote:
shubham
An anonymous person wrote:
parag
An anonymous person wrote:
pritesh
An anonymous person wrote:
ashesh
Sign Guestbook
Guestbook name: default_guestbook
switch
Logout