

E6156 - Topics in SW Engineering Management

Nestly: Cloud Rental Property Management System

*Zhang, Youcheng(yz4589), Yao, Xirui(xy2571), Chen, Shiying(sc5299),
Lin, Runze(rl3376), Wang, Isa(yw3886)*



TRANSCENDING DISCIPLINES, TRANSFORMING LIVES



COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

1. Introduction

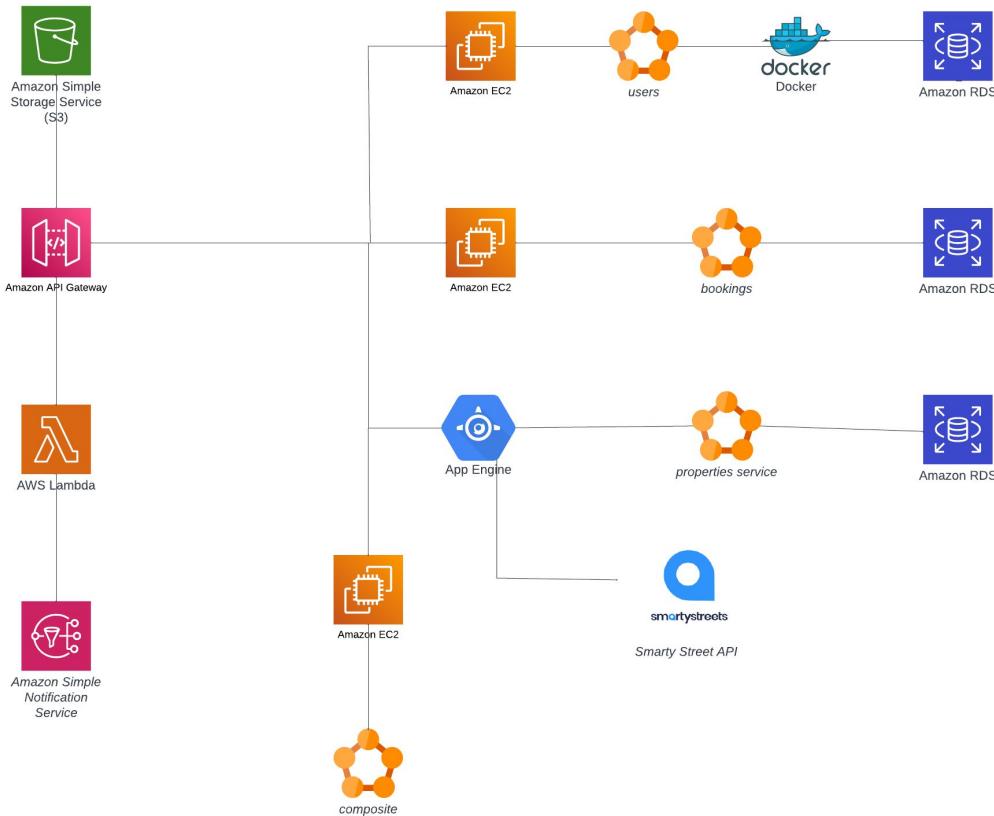
Nestly: Cloud Rental Property Management

- Facilitating seamless property listings and bookings for diverse users.
- Ensuring smooth interactions between hosts, guests, and administrators.
- Integrating multiple cloud services and maintaining architectural consistency across them.

2. 10 Required implementation demo

3. How our application works

Overall Structure



Enhance Microservices Implementation

Enhanced Microservices

finished on all 3 microservices, using Users as example

GET:
(getting all
here)

▲ 不安全 | ec2-3-144-182-9.us-east-2.compute.amazonaws.com:8012

Get All Users

Get All Users

Users Results

```
ID: 00000001, Username: Tarantula, Name: Morty Stokey, Email: mstokey0@slashdot.org, Credit: 294, OpenID: 9385195611, Role: host
ID: 00000002, Username: Bontebok, Name: Marguerite Farington, Email: mfarington1@adobe.com, Credit: 1885, OpenID: 0967253799, Role: guest
ID: 00000003, Username: Pheasant, common, Name: Amelia Bexon, Email: abexon2@jiathis.com, Credit: 2402, OpenID: 6571275465, Role: guest
ID: 00000004, Username: Azara's zorro, Name: Timmie Beine, Email: tbeine3@msu.edu, Credit: 4815, OpenID: 9514213866, Role: host
ID: 00000005, Username: Dusky rattlesnake, Name: Eileen Geary, Email: egeary4@t-online.de, Credit: 2599, OpenID: 3706445166, Role: host
ID: 00000006, Username: Weaver, chestnut, Name: Jemima Braghini, Email: jbraghini5@google.co.jp, Credit: 3503, OpenID: 3453605004, Role: host
ID: 00000007, Username: Gazer, sun, Name: Liv Toulmin, Email: ltoulmin6@1und1.de, Credit: 4341, OpenID: 6593544108, Role: host
ID: 00000008, Username: Shrike, southern white-crowned, Name: Darwin Adshead, Email: dadshead7@npr.org, Credit: 232, OpenID: 1350189928, Role: guest
ID: 00000009, Username: Arboral spiny rat, Name: Reynold Lambden, Email: rlambden8@java.com, Credit: 1715, OpenID: 2332988181, Role: guest
ID: 00000010, Username: Lemming, arctic, Name: Talia Reuven, Email: treuven9@wp.com, Credit: 4542, OpenID: 2251888098, Role: guest
ID: 00000011, Username: Cattle egret, Name: Vernen Gendrich, Email: vgendricha@cbslocal.com, Credit: 1471, OpenID: 1500125113, Role: host
ID: 00000012, Username: Grey fox, Name: Dedin Squelch, Email: dsquelchb@pagesperso-orange.fr, Credit: 1533, OpenID: 3610011556, Role: guest
ID: 00000013, Username: Green-winged macaw, Name: Adorne McIlvenny, Email: amcilvernny@europa.eu, Credit: 4283, OpenID: 5106976995, Role: host
ID: 00000014, Username: Emu, Name: Ines Ayton, Email: iaytond@unesco.org, Credit: 4160, OpenID: 7225331302, Role: guest
ID: 00000015, Username: Gull, pacific, Name: Janka Scotchmore, Email: jscotchmoree@flavors.me, Credit: 1881, OpenID: 6556520225, Role: guest
ID: 00000016, Username: Hawk, ferruginous, Name: Colin Main, Email: cmainf@who.int, Credit: 363, OpenID: 5467203681, Role: host
ID: 00000017, Username: Nilgai, Name: Celestine Lamps, Email: clamps9@imgur.com, Credit: 3242, OpenID: 4443379878, Role: host
ID: 00000018, Username: Pigeon, wood, Name: Miranda Crowch, Email: mcrowchh@typepad.com, Credit: 1898, OpenID: 9093904108, Role: guest
ID: 00000019, Username: Painted stork, Name: Hermine Blazic, Email: hblazici@dmoz.org, Credit: 706, OpenID: 7513278881, Role: host
ID: 00000020, Username: Arctic hare, Name: Charity Nelissen, Email: cnelissenj@fc2.com, Credit: 843, OpenID: 6810216715, Role: host
ID: 00000021, Username: Black-eyed bulbul, Name: Janith Riccetti, Email: jriccettik@deliciousdays.com, Credit: 522, OpenID: 6725080715, Role: guest
```

Enhance Microservices

PUT: (added a test user 'testtest' and the GET on username=testtest)

The screenshot shows a web application interface for managing users. At the top, there is a header bar with a warning icon and the URL "ec2-3-144-182-9.us-east-2.compute.amazonaws.com:8012". Below the header, there are several sections:

- Create User**: A form with fields for Username (testtest), First Name (firstname), Last Name (lastname), Email (test@gmail.com), and Credit (300). A dropdown menu shows "Guest" and a "Create User" button.
- Filter Users**: A form with fields for User ID, First Name, Last Name, Email, Credit (with dropdowns for "Credit", "Credit Greater Than", "Credit Less Than", and a range from 0 to 10), Role (dropdown with value 10), and an "Apply Filters" button.
- Get All Users**: A button labeled "Get All Users".
- Users Results**: A section displaying the result of a user search. It shows the ID, Username, Name, Email, Credit, OpenID, and Role for a user named "testtest". The text reads: "ID: 00000301, Username: testtest, Name: firstname lastname, Email: test@gmail.com, Credit: 300, OpenID: null, Role: guest".

Enhance Microservices

POST: (updated the added user using user id=00000301, username to 'changed!', and the GET on user id=00000301, showing username changed to 'changed!')

Users Results

...4-182-9.us-east-2.compute.amazonaws.com:8012 显示

User updated

Filter Users

00000301

Username

First Name

Last Name

Email

Credit

Credit Greater Than

Credit Less Than

Role

10

0

Apply Filters

Get All Users

Get All Users

Users Results

ID: 00000301, Username: **changed!**, Name: firstname lastname, Email: test@gmail.com, Credit: 300, OpenID: null, Role: guest

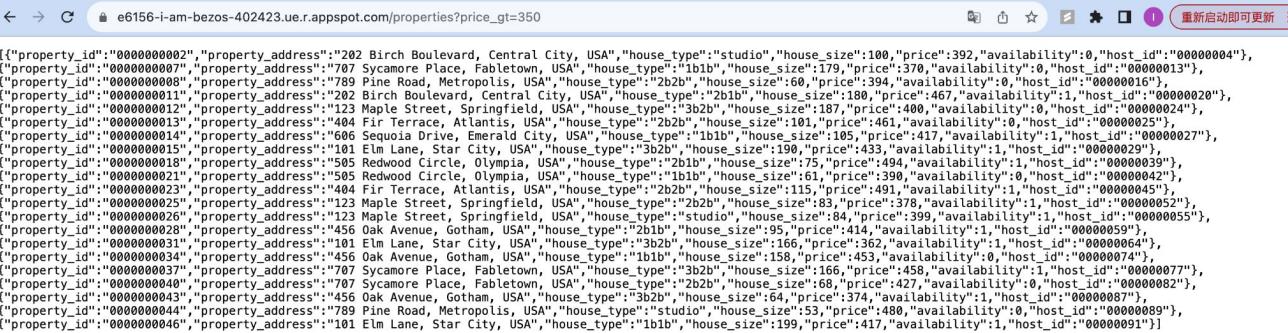
Enhance Microservices

DELETE
(DELETE user
id=00000301,
and GET user
id=00000301
returning “No
(such) users
were found”)

The screenshot shows a web application interface for managing users. At the top, there is a header bar with a URL (不安全 | ec2-3-144-182-9.us-east-2.compute.amazonaws.com:8012), browser controls, and a status message "...4-182-9.us-east-2.compute.amazonaws.com:8012 显示". Below the header, there are two main sections: "Users Results" and "Filter Users". The "Users Results" section displays the message "User deleted". The "Filter Users" section contains input fields for filtering users by ID, Username, First Name, Last Name, Email, Credit, Credit Greater Than, Credit Less Than, Role (set to 10), and a limit of 10. Below these sections are buttons for "Get All Users" and "Delete User". The "Get All Users" button is currently active. The "Delete User" section shows an input field with the value "00000301" and a "Delete User" button.

Enhance Microservices

Querying
(realized on
Properties)
(E.g.
price_gt=350,
return properties
with price greater
than 350)



```
[{"property_id": "0000000002", "property_address": "202 Birch Boulevard, Central City, USA", "house_type": "studio", "house_size": 100, "price": 392, "availability": 0, "host_id": "00000004"}, {"property_id": "0000000007", "property_address": "707 Sycamore Place, Fabletown, USA", "house_type": "1b1b", "house_size": 179, "price": 370, "availability": 0, "host_id": "00000013"}, {"property_id": "0000000008", "property_address": "789 Pine Road, Metropolis, USA", "house_type": "2b2b", "house_size": 60, "price": 394, "availability": 0, "host_id": "00000016"}, {"property_id": "0000000011", "property_address": "202 Birch Boulevard, Central City, USA", "house_type": "2b1b", "house_size": 180, "price": 467, "availability": 1, "host_id": "00000020"}, {"property_id": "0000000012", "property_address": "123 Maple Street, Springfield, USA", "house_type": "3b2b", "house_size": 187, "price": 400, "availability": 0, "host_id": "00000024"}, {"property_id": "0000000013", "property_address": "404 Fir Terrace, Atlantis, USA", "house_type": "2b2b", "house_size": 101, "price": 461, "availability": 0, "host_id": "00000025"}, {"property_id": "0000000014", "property_address": "606 Sequoia Drive, Emerald City, USA", "house_type": "1b1b", "house_size": 105, "price": 417, "availability": 1, "host_id": "00000027"}, {"property_id": "0000000015", "property_address": "101 Elm Lane, Star City, USA", "house_type": "3b2b", "house_size": 190, "price": 433, "availability": 1, "host_id": "00000029"}, {"property_id": "0000000018", "property_address": "505 Redwood Circle, Olympia, USA", "house_type": "2b1b", "house_size": 75, "price": 494, "availability": 1, "host_id": "00000039"}, {"property_id": "0000000021", "property_address": "505 Redwood Circle, Olympia, USA", "house_type": "1b1b", "house_size": 61, "price": 390, "availability": 0, "host_id": "00000042"}, {"property_id": "0000000023", "property_address": "404 Fir Terrace, Atlantis, USA", "house_type": "2b2b", "house_size": 115, "price": 491, "availability": 1, "host_id": "00000045"}, {"property_id": "0000000025", "property_address": "123 Maple Street, Springfield, USA", "house_type": "2b2b", "house_size": 83, "price": 378, "availability": 1, "host_id": "00000052"}, {"property_id": "0000000026", "property_address": "123 Maple Street, Springfield, USA", "house_type": "studio", "house_size": 84, "price": 399, "availability": 1, "host_id": "00000055"}, {"property_id": "0000000028", "property_address": "456 Oak Avenue, Gotham, USA", "house_type": "2b1b", "house_size": 95, "price": 414, "availability": 1, "host_id": "00000059"}, {"property_id": "0000000031", "property_address": "101 Elm Lane, Star City, USA", "house_type": "3b2b", "house_size": 166, "price": 362, "availability": 1, "host_id": "00000064"}, {"property_id": "0000000034", "property_address": "456 Oak Avenue, Gotham, USA", "house_type": "1b1b", "house_size": 158, "price": 453, "availability": 0, "host_id": "00000074"}, {"property_id": "0000000037", "property_address": "707 Sycamore Place, Fabletown, USA", "house_type": "3b2b", "house_size": 166, "price": 458, "availability": 1, "host_id": "00000077"}, {"property_id": "0000000040", "property_address": "707 Sycamore Place, Fabletown, USA", "house_type": "2b2b", "house_size": 68, "price": 427, "availability": 0, "host_id": "00000082"}, {"property_id": "0000000043", "property_address": "456 Oak Avenue, Gotham, USA", "house_type": "3b2b", "house_size": 64, "price": 374, "availability": 1, "host_id": "00000087"}, {"property_id": "0000000044", "property_address": "789 Pine Road, Metropolis, USA", "house_type": "studio", "house_size": 53, "price": 480, "availability": 0, "host_id": "00000089"}, {"property_id": "0000000046", "property_address": "101 Elm Lane, Star City, USA", "house_type": "1b1b", "house_size": 199, "price": 417, "availability": 1, "host_id": "00000091"}]
```

Enhance Microservices

Querying
(realized on
Properties)
(E.g. with front
end, house_type
= 1b1b)

Filter Properties

Property ID	Property Address	1b1b	House size	House size Greater Than
House size Less Than	Price	Price Greater Than	Price Less Than	Availability (true/false)
Host ID	Limit	Offset	Apply Filters	

Get All Properties

[Get All Properties](#)

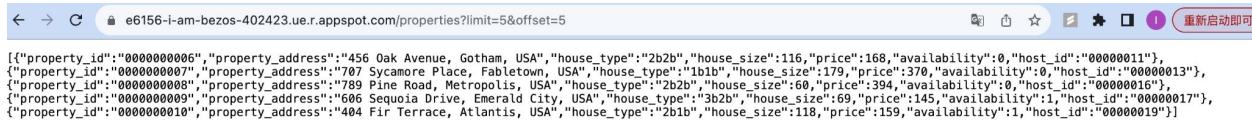
Properties Results

ID: 000000001, Address: 202 Birch Boulevard, Central City, USA, Type: 1b1b, Size: 102, Price: 135, Availability: 0, Host ID: 00000001
ID: 000000007, Address: 707 Sycamore Place, Fabletown, USA, Type: 1b1b, Size: 179, Price: 370, Availability: 0, Host ID: 00000013
ID: 000000014, Address: 606 Sequoia Drive, Emerald City, USA, Type: 1b1b, Size: 105, Price: 417, Availability: 1, Host ID: 00000027
ID: 000000016, Address: 123 Maple Street, Springfield, USA, Type: 1b1b, Size: 81, Price: 176, Availability: 0, Host ID: 00000035
ID: 000000021, Address: 505 Redwood Circle, Olympia, USA, Type: 1b1b, Size: 61, Price: 390, Availability: 0, Host ID: 00000042
ID: 000000022, Address: 202 Birch Boulevard, Central City, USA, Type: 1b1b, Size: 137, Price: 316, Availability: 1, Host ID: 00000044
ID: 000000034, Address: 456 Oak Avenue, Gotham, USA, Type: 1b1b, Size: 158, Price: 453, Availability: 0, Host ID: 00000074
ID: 000000038, Address: 404 Fir Terrace, Atlantis, USA, Type: 1b1b, Size: 127, Price: 242, Availability: 1, Host ID: 00000078
ID: 000000046, Address: 101 Elm Lane, Star City, USA, Type: 1b1b, Size: 199, Price: 417, Availability: 1, Host ID: 00000091
ID: 000000051, Address: 358 Broome Street New York, NY, Type: 1b1b, Size: 1, Price: 1, Availability: 1, Host ID: 00000002

Enhance Microservices

finished on all 3 microservices, using Users as example

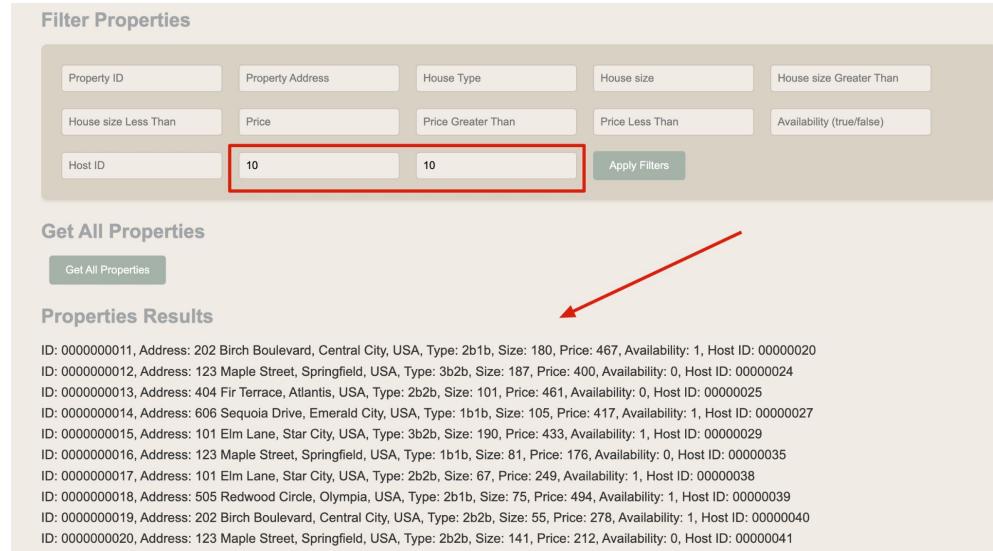
(E.g. Limit = 5 & Offset = 5 -> we got IDs from 6 to 10, instead of from 1 to 50)



```
[{"property_id": "0000000006", "property_address": "456 Oak Avenue, Gotham, USA", "house_type": "2b2b", "house_size": 116, "price": 168, "availability": 0, "host_id": "00000011"}, {"property_id": "0000000007", "property_address": "705 Sycamore Place, Fabletown, USA", "house_type": "1b1b", "house_size": 179, "price": 370, "availability": 0, "host_id": "00000013"}, {"property_id": "0000000008", "property_address": "789 Pine Road, Metropolis, USA", "house_type": "2b2b", "house_size": 60, "price": 394, "availability": 0, "host_id": "00000010"}, {"property_id": "0000000009", "property_address": "606 Sequoia Drive, Emerald City, USA", "house_type": "3b2b", "house_size": 69, "price": 145, "availability": 1, "host_id": "00000017"}, {"property_id": "0000000010", "property_address": "404 Fir Terrace, Atlantis, USA", "house_type": "2b1b", "house_size": 118, "price": 159, "availability": 1, "host_id": "00000019"}]
```

(E.g. with front end, Limit = 10 & Offset = 10 -> we got IDs from 11 to 20, instead of from 1 to 50)

Pagination &
offset (realized
on Properties)



Filter Properties

Property ID	Property Address	House Type	House size	House size Greater Than
House size Less Than	Price	Price Greater Than	Price Less Than	Availability (true/false)
Host ID	10	10	Apply Filters	

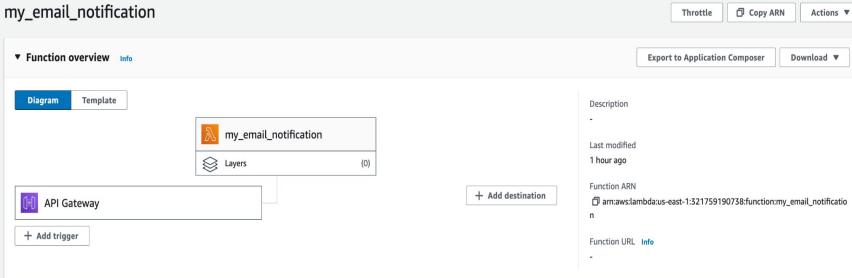
Get All Properties

Properties Results

ID: 0000000011, Address: 202 Birch Boulevard, Central City, USA, Type: 2b1b, Size: 180, Price: 467, Availability: 1, Host ID: 00000020
ID: 0000000012, Address: 123 Maple Street, Springfield, USA, Type: 3b2b, Size: 187, Price: 400, Availability: 0, Host ID: 00000024
ID: 0000000013, Address: 404 Fir Terrace, Atlantis, USA, Type: 2b2b, Size: 101, Price: 461, Availability: 0, Host ID: 00000025
ID: 0000000014, Address: 606 Sequoia Drive, Emerald City, USA, Type: 1b1b, Size: 105, Price: 417, Availability: 1, Host ID: 00000027
ID: 0000000015, Address: 101 Elm Lane, Star City, USA, Type: 3b2b, Size: 190, Price: 433, Availability: 1, Host ID: 00000029
ID: 0000000016, Address: 123 Maple Street, Springfield, USA, Type: 1b1b, Size: 81, Price: 176, Availability: 0, Host ID: 00000035
ID: 0000000017, Address: 101 Elm Lane, Star City, USA, Type: 2b2b, Size: 67, Price: 249, Availability: 1, Host ID: 00000038
ID: 0000000018, Address: 505 Redwood Circle, Olympia, USA, Type: 2b1b, Size: 75, Price: 494, Availability: 1, Host ID: 00000039
ID: 0000000019, Address: 202 Birch Boulevard, Central City, USA, Type: 2b2b, Size: 55, Price: 278, Availability: 1, Host ID: 00000040
ID: 0000000020, Address: 123 Maple Street, Springfield, USA, Type: 2b2b, Size: 141, Price: 212, Availability: 0, Host ID: 00000041

Events, Notifications, Pub/Sub

Events, Notifications, Pub/Sub



Code | Test | Monitor | Configuration | Aliases | Versions

Code source Info

Upload from ▾

File Edit Find View Go Tools Window Test Deploy

Go to Anything (⌘ P) lambda_function Environment Var

my_email_notifications λ lambda_function.py

```
1 import boto3
2 import http.client
3 import json
4
5 def lambda_handler(event, context):
6     # Parse the event
7     booking_data = json.loads(event['body']) # assuming the event body is JSON
8
9     # Prepare the request to EC2 endpoint
10    ec2_endpoint = "ec2-3-144-93-114.us-east-2.compute.amazonaws.com"
11    port = 8080
12    url = "/bookings"
13    headers = {'Content-type': 'application/json'}
14
15    # Send an HTTP request to the EC2 endpoint
16    connection = http.client.HTTPConnection(ec2_endpoint, port)
17    connection.request('POST', url, json.dumps(booking_data), headers)
18
19    response = connection.getresponse()
20
21    if response.status == 200:
22        # Read and parse the response from EC2
23        response_data = response.read().decode()
24
25        ec2_response_data = json.loads(response_data)
26    except json.JSONDecodeError:
27        # Handle the case where the response is not JSON
28        connection.close()
29        return {
30            'statusCode': 500,
31            'body': json.dumps('Invalid JSON from EC2')
32        }
33
34
35    # Send an email (using SNS or SES)
36    sns_client = boto3.client('sns')
37    snsArn = 'arn:aws:sns:us-east-1:321759190738:nestlyNotification'
38    message = "Dear host, Great news! We are thrilled to inform you that your property has been successfully booked through Nestly. Details: " + ec2_response_data["details"]
39
40    sns_response = sns_client.publish(
41        TopicArn=snsArn,
42        Message=message,
43        Subject='New Booking Alert: Your Property has been Booked on Nestly!'
44    )
45
46
47    connection.close()
48    return {
49        'statusCode': 200,
50        'body': json.dumps('Email sent successfully')
51    }
52
53    # Handle errors from EC2 request
54    connection.close()
55    return {
56        'statusCode': response.status,
57        'body': json.dumps('Error communicating with EC2')
58    }
```

1:1 Python Spaces: 4

Composition/Aggregators

Composition/Aggregators

Sync: (designated order always)

```
INFO: Started server process [14318]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://0.0.0.0:8012 (Press CTRL+C to quit)
INFO: 127.0.0.1:52368 - "GET /docs HTTP/1.1" 200 OK
INFO: 127.0.0.1:52368 - "GET /openapi.json HTTP/1.1" 200 OK
users service has returned a response (in sync)
properties service has returned a response (in sync)
bookings service has returned a response (in sync)
INFO: 127.0.0.1:52369 - "GET /sync-aggregator HTTP/1.1" 200 OK
users service has returned a response (in sync)
properties service has returned a response (in sync)
bookings service has returned a response (in sync)
```

Async: different orders

```
INFO: 127.0.0.1:52408 - "GET /async-aggregator HTTP/1.1" 200 OK
users service has returned a response (in async)
properties service has returned a response (in async)
bookings service has returned a response (in async)
INFO: 127.0.0.1:52415 - "GET /async-aggregator HTTP/1.1" 200 OK
bookings service has returned a response (in async)
properties service has returned a response (in async)
users service has returned a response (in async)
INFO: 127.0.0.1:52420 - "GET /async-aggregator HTTP/1.1" 200 OK
bookings service has returned a response (in async)
properties service has returned a response (in async)
users service has returned a response (in async)
INFO: 127.0.0.1:52424 - "GET /async-aggregator HTTP/1.1" 200 OK
users service has returned a response (in async)
bookings service has returned a response (in async)
properties service has returned a response (in async)
INFO: 127.0.0.1:52428 - "GET /async-aggregator HTTP/1.1" 200 OK
bookings service has returned a response (in async)
users service has returned a response (in async)
properties service has returned a response (in async)
INFO: 127.0.0.1:52436 - "GET /async-aggregator HTTP/1.1" 200 OK
bookings service has returned a response (in async)
users service has returned a response (in async)
properties service has returned a response (in async)
INFO: 127.0.0.1:52440 - "GET /async-aggregator HTTP/1.1" 200 OK
users service has returned a response (in async)
bookings service has returned a response (in async)
users service has returned a response (in async)
properties service has returned a response (in async)
INFO: 127.0.0.1:52449 - "GET /async-aggregator HTTP/1.1" 200 OK
properties service has returned a response (in async)
bookings service has returned a response (in async)
users service has returned a response (in async)
INFO: 127.0.0.1:52453 - "GET /async-aggregator HTTP/1.1" 200 OK
```

API Gateway

Resources

Create resource

/

/bookings

DELETE

GET

OPTIONS

POST

PUT

/{{booking_id}}

DELETE

PUT

/properties

DELETE

GET

OPTIONS

POST

PUT

/{{property_id}}

DELETE

PUT

/users

DELETE

GET

OPTIONS

POST

PUT

/{{user_id}}

DELETE

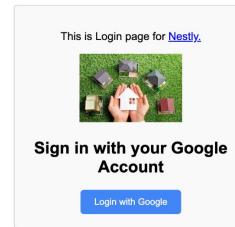
PUT



Single Sign-On (SSO)

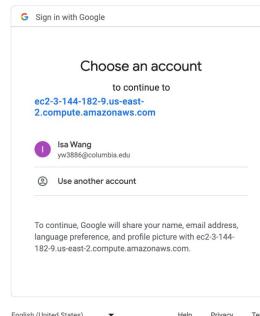
Single Sign-On (SSO)

← → C 不安全 | https://ec2-3-144-182-9.us-east-2.compute.amazonaws.com



```
INFO: 209.2.224.170:0 - "GET /login HTTP/1.1" 200 OK
INFO: 209.2.224.170:0 - "GET /rent.jpeg HTTP/1.1" 200 OK
INFO: 209.2.224.170:0 - "GET /auth/login HTTP/1.1" 303 See Other
Request = <starlette.requests.Request object at 0x7f861454cf10>
URL = http://localhost:8012/auth/callback?code=4%2F0AfJohXm0_faSL0ANyP5QDe8xzn28smd8nid+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com%2Foauth2%2Fuserinfo&prompt=consent
INFO: 209.2.224.170:0 - "GET /auth/callback?code=4%2F0AfJohXm0_faSL0ANyP5QDe8xzn28smd8e+openid+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com%2Foauth2%2Fuserinfo&prompt=consent HTTP/1.1" 200 OK
```

← → C ✕ 不安全 | https://ec2-3-144-182-9.us-east-2.compute.amazonaws.com/auth/callback?code=4%2F0AfJohXm0_faSL0ANyP5QDe8xzn28smd8nid+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com%2Foauth2%2Fuserinfo&prompt=consent



User Information



OpenID: 100214969192775684788

Email: yw3886@columbia.edu

First Name: Isa

Last Name: Wang

Display Name: Isa Wang

[Login as Host](#) [Login as Guest](#) [Login as Admin](#)

External API

External API

The image displays two screenshots of a web-based property management system. Both screenshots show a header bar with the URL `e6156-i-am-bezos-402423.ue.r.appspot.com` and a toolbar with various icons.

Screenshot 1 (Successful Creation): A modal dialog box is open, displaying the message "Property updated successfully" with a blue "确定" (Confirm) button. The main interface shows a "Create Property" form with the following fields:

- Address: 500 W 120th St, New York, NY
- Unit: 2b3b
- Bedrooms: 400
- Bathrooms: 5000
- Status: Not Available
- Phone: 00000001
- Create Property button

Screenshot 2 (Failed Creation): A modal dialog box is open, displaying the message "Invalid address" with a blue "确定" (Confirm) button. A red arrow points from the text "Invalid address" to the "Address" field in the main form. The "Address" field is highlighted with a red border. The main interface shows the same "Create Property" form as in the first screenshot, but with the address field now containing the text "I know this will fail".

Middleware

Middleware

The screenshot shows a terminal window on the AWS CloudWatch interface. At the top, there are navigation links for 'aws' (with the AWS logo), 'Services' (with a grid icon), and a search bar with placeholder text '[Option+S]'. The main area displays a command-line session.

```
From github.com:Runze-Lin/Micoservice-Bookings-and-Review
  82f7800..917fac4  main      -> origin/main
Updating 82f7800..917fac4
Fast-forward
 app.py                  |  2 +-+
 static/user_booking_management.js |  5 +++--+
 2 files changed, 4 insertions(+), 3 deletions(-)
[ec2-user@ip-172-31-10-119 Micoservice-Bookings-and-Review]$ python3 app.py
Database connected successfully!
INFO:     Started server process [905790]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8012 (Press CTRL+C to quit)
Request: GET http://ec2-3-144-93-114.us-east-2.compute.amazonaws.com:8012/
Response: Status 307
INFO:     209.2.224.170:51657 - "GET / HTTP/1.1" 307 Temporary Redirect
Request: GET http://ec2-3-144-93-114.us-east-2.compute.amazonaws.com:8012/clientType
Response: Status 307
INFO:     209.2.224.170:51745 - "GET /clientType HTTP/1.1" 307 Temporary Redirect
Request: GET http://ec2-3-144-93-114.us-east-2.compute.amazonaws.com:8012/host
Response: Status 307
INFO:     209.2.224.170:51745 - "GET /host HTTP/1.1" 307 Temporary Redirect
Request: GET http://ec2-3-144-93-114.us-east-2.compute.amazonaws.com:8012/booking_management_host/00000001
Response: Status 307
INFO:     209.2.224.170:51745 - "GET /booking_management_host/00000001 HTTP/1.1" 307 Temporary Redirect
Request: GET http://ec2-3-144-93-114.us-east-2.compute.amazonaws.com:8012/bookings?host_id=00000001
Response: Status 200
INFO:     209.2.224.170:51767 - "GET /bookings?host_id=00000001 HTTP/1.1" 200 OK
```

CI/CD

CI/CD

Jobs:

1. Build Job:

- **Environment:** It runs on the latest Ubuntu runner provided by GitHub Actions.
- **Steps:**
 - Checkout: actions/checkout@v2.
 - Python Setup: Sets up Python 3.8
 - Install Dependencies: Upgrades pip and installs pytest along with the packages specified in requirements.txt

2. Deploy Job:

- **Dependency:** It needs the build job to complete first.
- **Condition:** It only runs when a push event happens on the main branch.
- **Environment:** It also runs on the latest Ubuntu runner.
- **Steps:**

- Checkout: Checks out the code.
- Install Dependencies
- SSH Key Setup
- SSH Keyscan
- Deploy

The image shows two screenshots of the GitHub Actions interface. The top screenshot displays the 'Actions' tab for a pipeline named 'CI/CD Pipeline'. It shows a single job named 'add logging middleware #30' which has completed successfully. The bottom screenshot shows the 'Actions' tab for the same repository, listing several workflow runs. The runs are ordered by age, with the most recent being 'add logging middleware' (main branch, 1 hour ago) and the oldest being 'Update main.yml' (main branch, yesterday).

Workflow Run	Event	Status	Branch	Actor	Time Ago
add logging middleware	CI/CD Pipeline #30: Commit da918fd pushed by IsaWang-05	Success	main	IsaWang-05	1 hour ago
update host_id insert	CI/CD Pipeline #29: Commit zcdba47 pushed by IsaWang-05	Success	main	IsaWang-05	2 hours ago
minor bug fix	CI/CD Pipeline #28: Commit 9014698 pushed by IsaWang-05	Success	main	IsaWang-05	8 hours ago
removed edit for user	CI/CD Pipeline #27: Commit ddb22303 pushed by xyao997	Success	main	xyao997	8 hours ago
update booking.management page and client	CI/CD Pipeline #26: Commit b9389df pushed by xyao997	Success	main	xyao997	16 hours ago
update host page and host_order_manage...	CI/CD Pipeline #25: Commit 529f2ff6 pushed by xyao997	Success	main	xyao997	17 hours ago
Update main.yml	CI/CD Pipeline #24: Commit 41d1606 pushed by sychen010223	Success	main	sychen010223	yesterday

Infrastructure As Code

Infrastructure As Code

Use Terraform to create and initialize an EC2 instance in the terminal (`./terraform apply`), then destroy it in the terminal (`./terraform destroy`).

Create:

```
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Still creating... [20s elapsed]
aws_instance.example: Still creating... [30s elapsed]
aws_instance.example: Creation complete after 33s [id=i-05d80bde8aa335c24]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
(base)
LilyC@LAPTOP-PV8BHP7E MINGW64 /d/CU/COMS6156/FinalProj/Micoservice-Bookings (
```

Destroy:

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure for "MICOSERVICE-BOOKINGS" containing files like ".github", ".terraform", "static", "__init__.py", ".terraform.lock.hcl", "app.py", "bookings.py", "LICENSE", "main.tf", "README.md", "requirements.txt", "terraform.exe", "terraform.tfstate", "terraform.tfstate.backup", and "test.txt".
- Code Editor:** Displays the "main.tf" file with the following Terraform code:

```
provider "aws" {
  region = "us-west-2"
}

resource "aws_instance" "example" {
  ami           = "ami-0c5f377539eaebf12"
  instance_type = "t2.micro"

  tags = {
    Name = "ExampleInstance"
  }
}
```
- Terminal:** Shows the output of the Terraform command "destroy".

```
aws_instance.example: Still destroying... [id=i-0005baa3a749eeff27, 40s elapsed]
aws_instance.example: Still destroying... [id=i-0005baa3a749eeff27, 50s elapsed]
aws_instance.example: Still destroying... [id=i-0005baa3a749eeff27, 1m0s elapsed]
aws_instance.example: Still destroying... [id=i-0005baa3a749eeff27, 1m20s elapsed]
aws_instance.example: Still destroying... [id=i-0005baa3a749eeff27, 1m30s elapsed]
aws_instance.example: Still destroying... [id=i-0005baa3a749eeff27, 1m40s elapsed]
aws_instance.example: Destruction complete after 1m42s

Destroy complete! Resources: 1 destroyed.
(base)
LilyC@LAPTOP-PV8BHP7E MINGW64 /d/CU/COMS6156/FinalProj/Micoservice-Bookings (main)
```

GraphQL

GraphQL

GraphQL to view
the users table
Query the id,
username, email.

```
query{
  allUsers{
    id
    username
    email
  }
}
```

```
{
  "data": {
    "allUsers": [
      {
        "id": "00000001",
        "username": "Tarantula",
        "email": "mstokey@slashdot.org"
      },
      {
        "id": "00000002",
        "username": "Bontebok",
        "email": "mfarington1@adobe.com"
      },
      {
        "id": "00000003",
        "username": "Pheasant, common",
        "email": "abexon2@jiathis.com"
      },
      {
        "id": "00000004",
        "username": "Azara's zorro",
        "email": "tbeine3@msu.edu"
      },
      {
        "id": "00000005",
        "username": "Dusky rattlesnake",
        "email": "egeary4@t-online.de"
      },
      {
        "id": "00000006",
        "username": "Weaver, chestnut",
        "email": "jbraghini5@google.co.jp"
      },
      {
        "id": "00000007",
        "username": "G  z  r, sun",
        "email": "ltoulmin6@lndl.de"
      },
      {
        "id": "00000008",
        "username": "Shrike, southern white-crowned",
        "email": "shrike8@shrike8.com"
      }
    ]
  }
}
```

Thank you for your listening!