Project Nestly

E6156 - Topics in SW Engineering Management: Cloud Rental Property Management System

Focal Point: Zhang, Youcheng(yz4589)
Team Members:

Yao, Xirui, xy2571

Chen, Shiying, sc5299

Lin, Runze, rl3376

Wang, Isa, yw3886

Demo video link

Demo video one piece:
https://drive.google.com/drive/folders/1NHVRm7xkr4ZcbKqtPqhoU3HXuEmL0UWI

Demo video in parts (intro + 10 required parts + business logic)
https://drive.google.com/drive/folders/1fI35cs0cxPva7Q74Hgsx2wI9x5aow5Ey

Entire code zip link

https://drive.google.com/drive/folders/1N-gnaxZV_EN-9F4rD7GrQ-IAM9Me81n3

Introduction

Nestly: Cloud Rental Property Management, the complexities are numerous:

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

Nestly, a cloud-native platform tailored to revolutionize the rental property management landscape. It's not just a tool but an embodiment of efficiency, designed to cater to the varied needs of property listing, management, and booking.

Personas and Roles

There are 3 personas:

1. Guest
    a. Unregistered guest
    b. Registered guest
2. Host/Landlord
3. Customer Service (Admin)

Key Features

An Unregistered Guest can:

1. Search for properties based on address, house type, house size, availability, host_id, and price.

A Registered Guest can:

1. Register and log in using Google SSO
2. Search for properties based on address, house type, house size, availability, host_id, and price.
3. Create and cancel bookings for available properties
4. View their own bookings

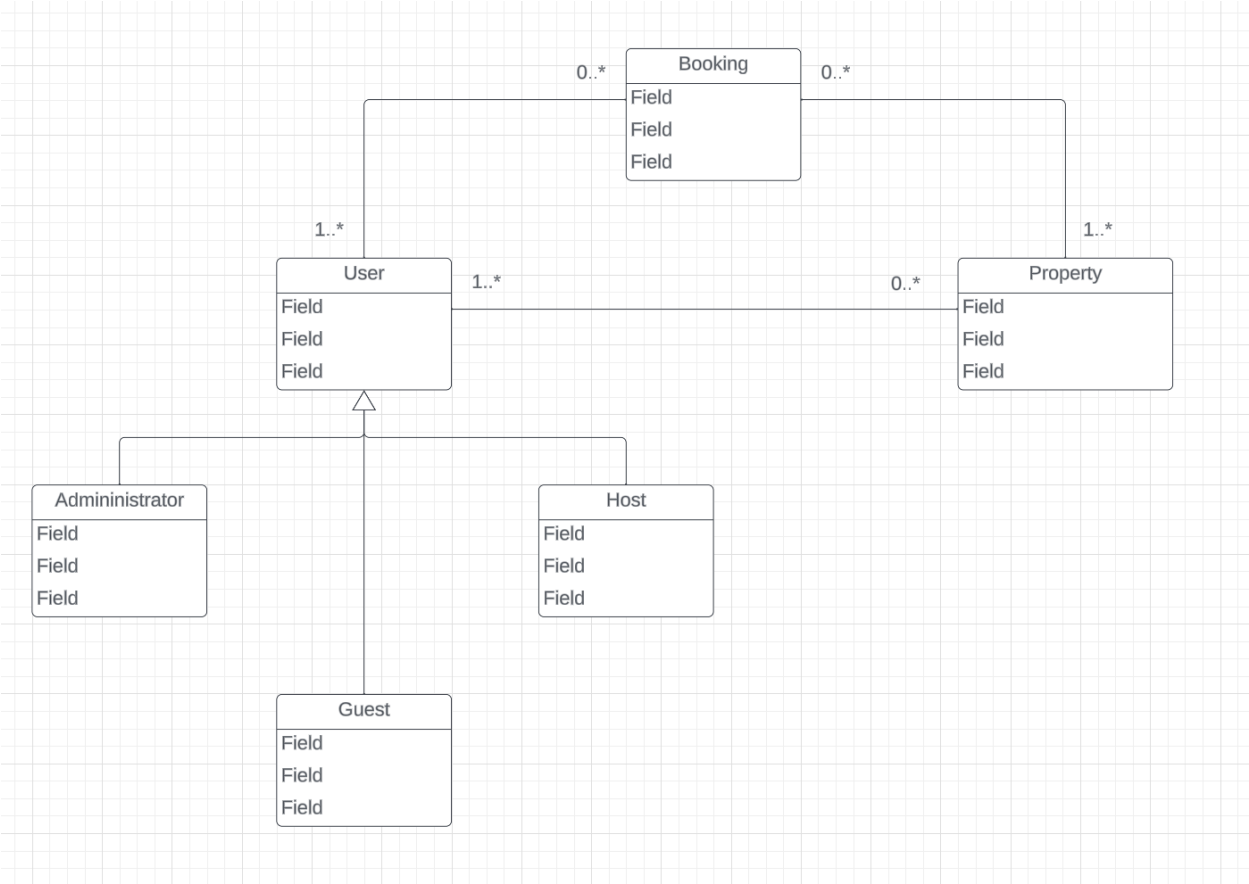
A Host can:

1. Register and log in using Google SSO
2. List their properties, availability, and pricing.
3. Receive notifications for bookings made by guests for their property.
4. Update the availability and pricing of their property as needed.
5. View their own properties
6. View their own bookings

An Administrator can:

1. Register and log in using Google SSO
2. Manage all property listings. Get, create, update, and remove property listings.
3. Manage all user information. Get, create, update, and delete users.
4. Manage all bookings. Get, create, update, and remove bookings

Domain Model



The domain model outlines the main entities and their relationships within the system. At its core is the "User" entity, which can be an admin, a host, an unregistered guest, or a registered guest. Users can make and own multiple bookings and properties. "Booking" represents the reservation of a property, while "Property" signifies accommodations available for booking. Admins have overarching control, hosts offer properties, and guests book them. The relationships between these entities highlight the interactions within the system like users making bookings or owning properties.

Resource Paths

/api/docs: OpenAPI documentation.

Users:

- Guests (Registered):
    - /api/users/{user_id}

- Host:
    - /api/users/{user_id}

- Admin:
    - /api/users (all users) (filter & pagination enabled)
    - /api/users/{user_id}

Property:

- /api/properties (filter & pagination enabled)
- /api/properties/{property_id}
- /api/properties/search (for guests)
- /api/properties/host_property_management/{host_id} (for host)

Booking:

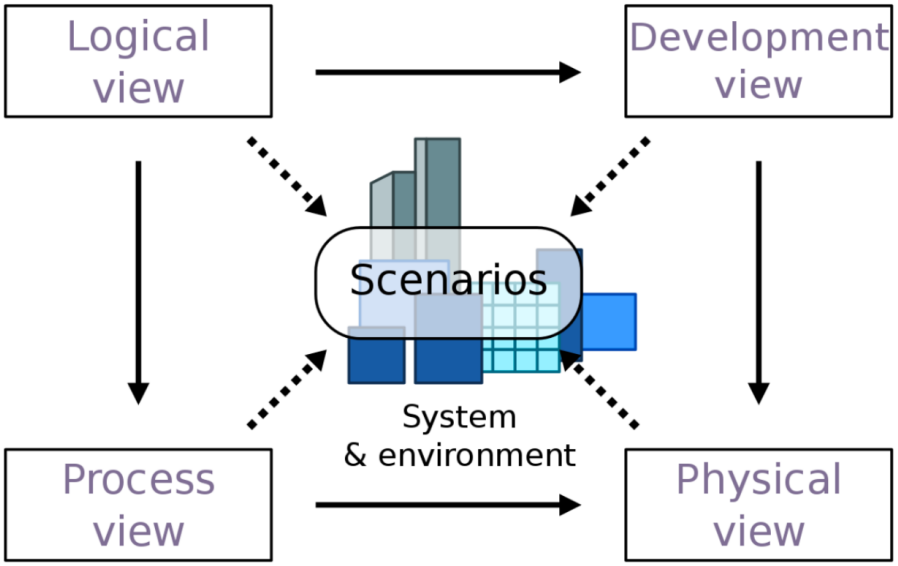
- /api/bookings (filter & pagination enabled)
- /api/bookings/{booking-id}
- /api/bookings/booking_management_host/{host_id}
- /api/bookings/user_booking_management/{host_id}

Composite:

- /api/clientType
- /api/user (stands for guest here)
- /api/host

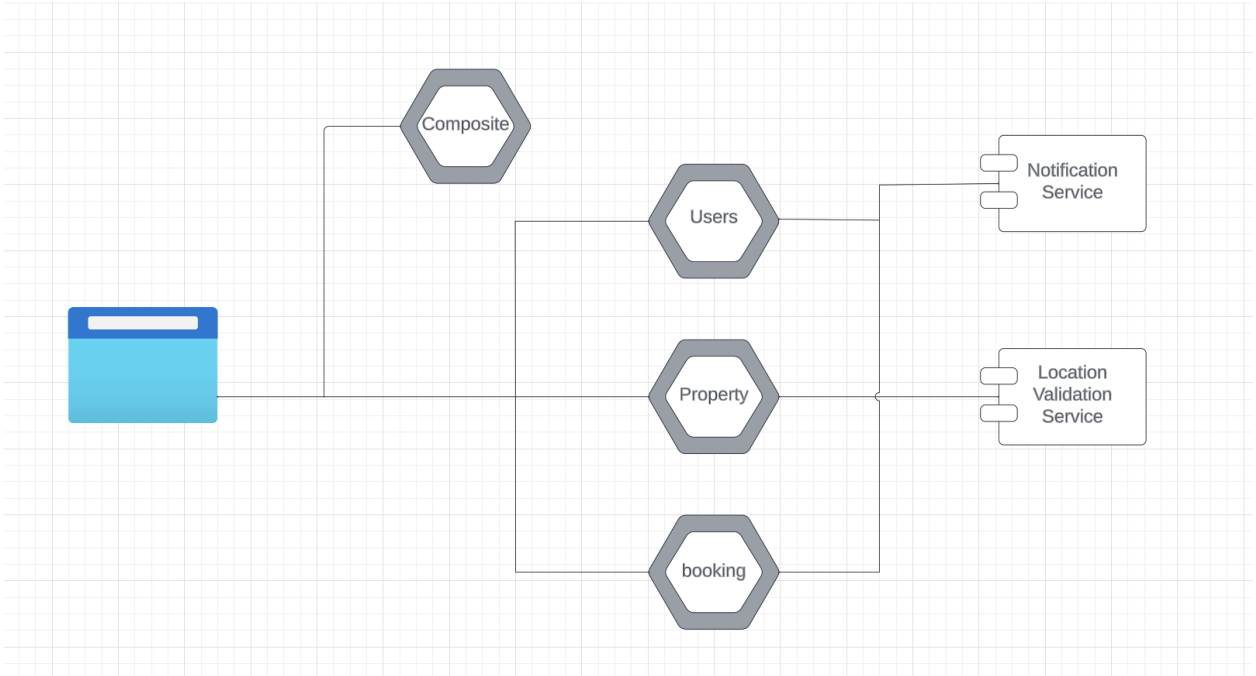
Architecture

Our team uses the subset of the approach: logical view and physical view. And will be more complete over time.



Logical View

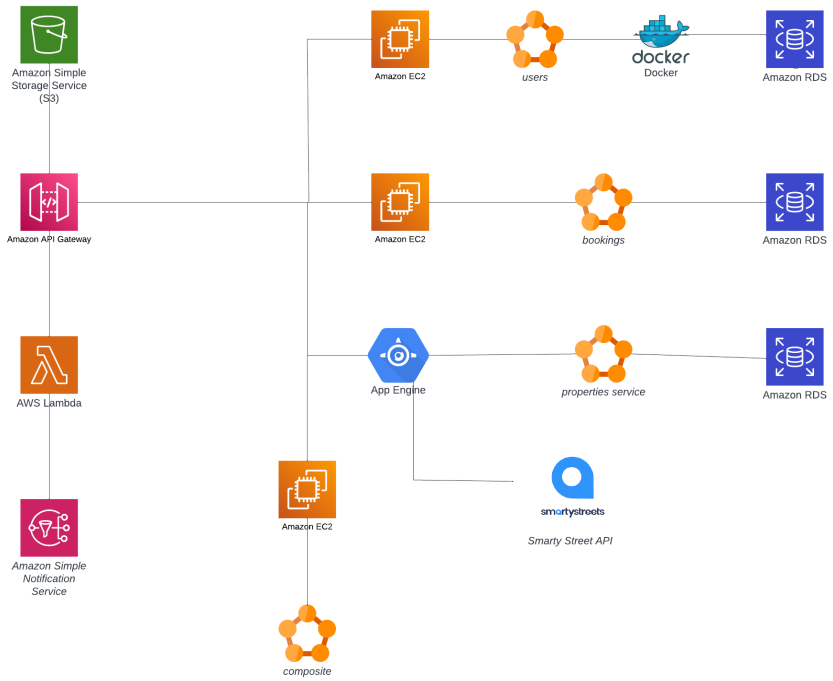
A basic concept in our logical view is following HexAgonal Architecture.



1. Composite: Acts as an orchestrator or aggregator, consolidating data and operations from other microservices to provide comprehensive functionality. Mostly in charge of redirecting to their related functions for different roles (host/guest).
2. User Management Service:
    a. Manage registration and login functionalities for all user types other than Unregistered Guests: (Registered) Guest, Host, and Administrator
    b. Manage role-based access control
    c. Allow update & delete of users
3. Property Management Service:
    a. Allow hosts to list, update, and manage their properties.
    b. Handle searches for properties based on various criteria such as location, availability, and price
    c. Manage the availability and pricing updates for properties.
4. Booking Service:
    a. Handle the entire booking process, including creating & canceling(deleting) process
    b. Send email notifications to hosts when their properties get booked.
    c. Allow Admin to mange bookings (E.g. update the guest_id and property_id linked with a bookings)

Physical View

This is the draft, in-progress physical view of the first few milestones.



Three microservices:

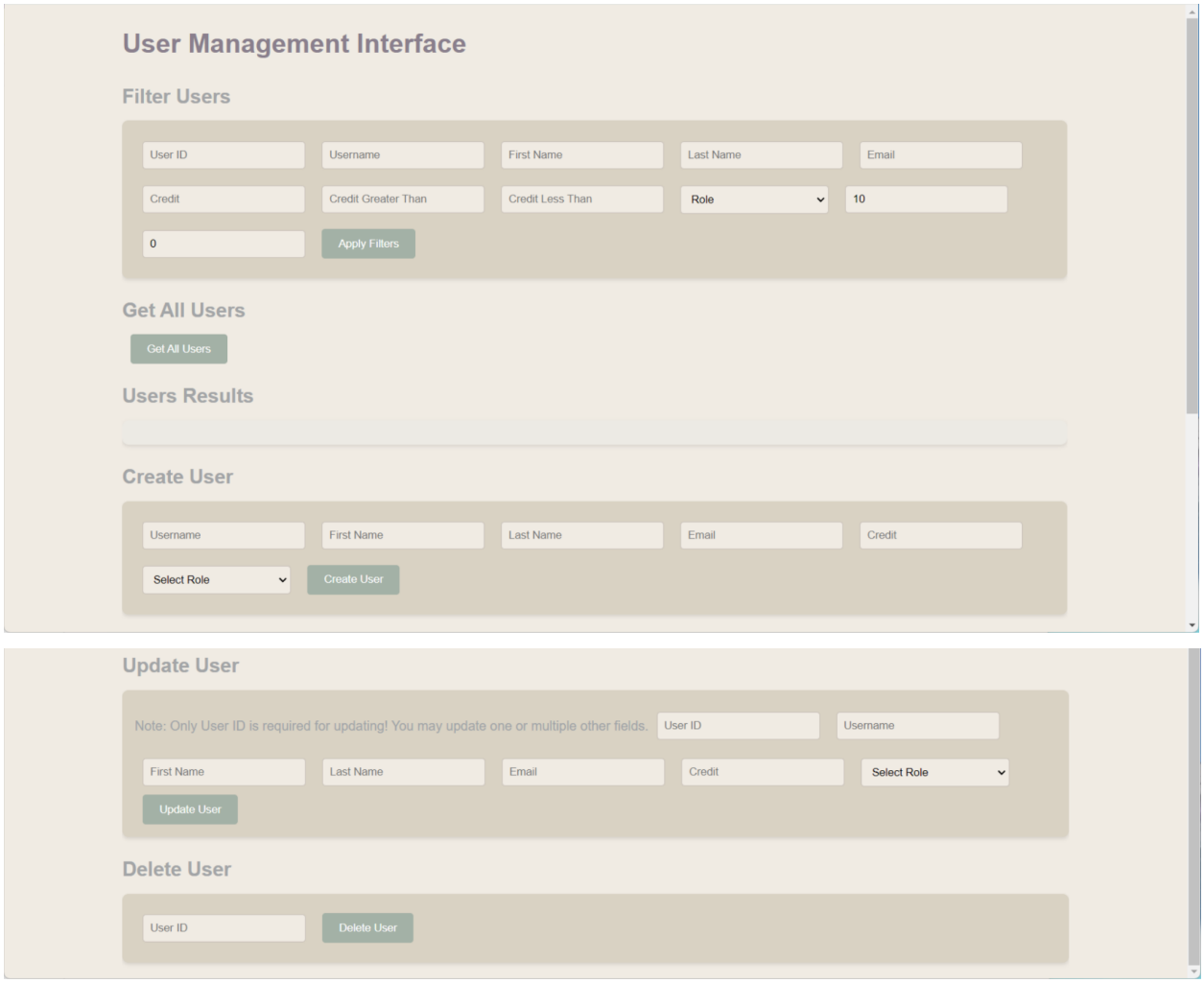
1. User Management Service on EC2 with Docker
    a. Database interaction with Amazon RDS
2. Property Management Service on GCP:
    a. Database interaction with Amazon RDS
3. Booking & Review Service on EC2 without Docker
    a. Database interaction with Amazon RDS

Codebase

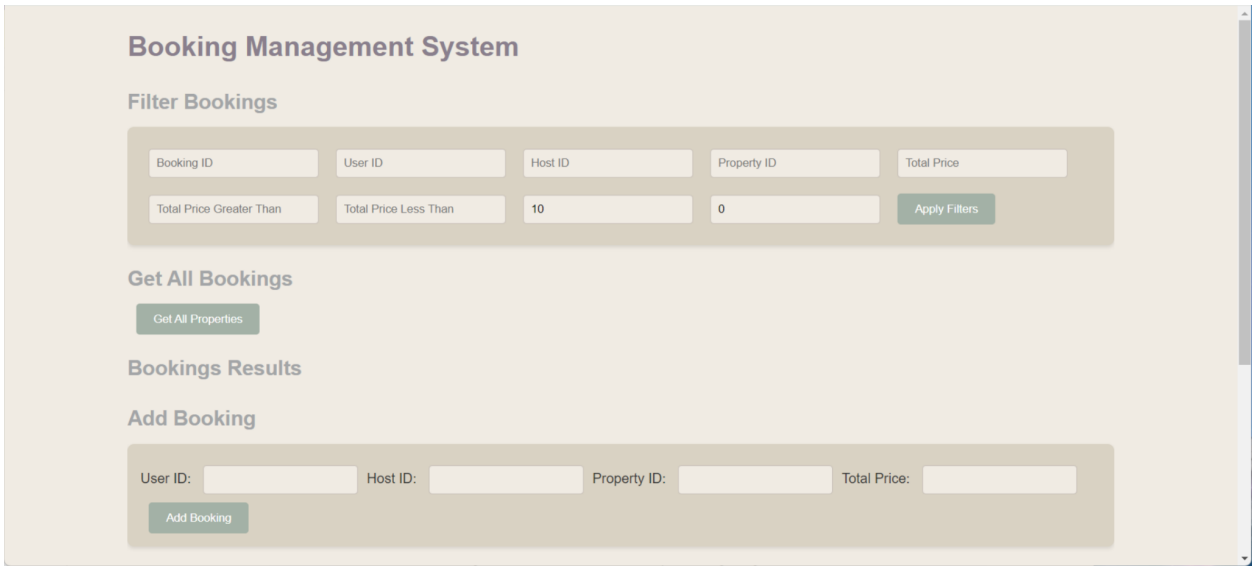
https://github.com/Runze-Lin/Micoservice-User
https://github.com/Runze-Lin/Micoservice-Bookings-and-Review
https://github.com/Runze-Lin/Micoservice-Property
https://github.com/Runze-Lin/Composite
https://github.com/users/sychen010223/projects/1

Links to our deployment

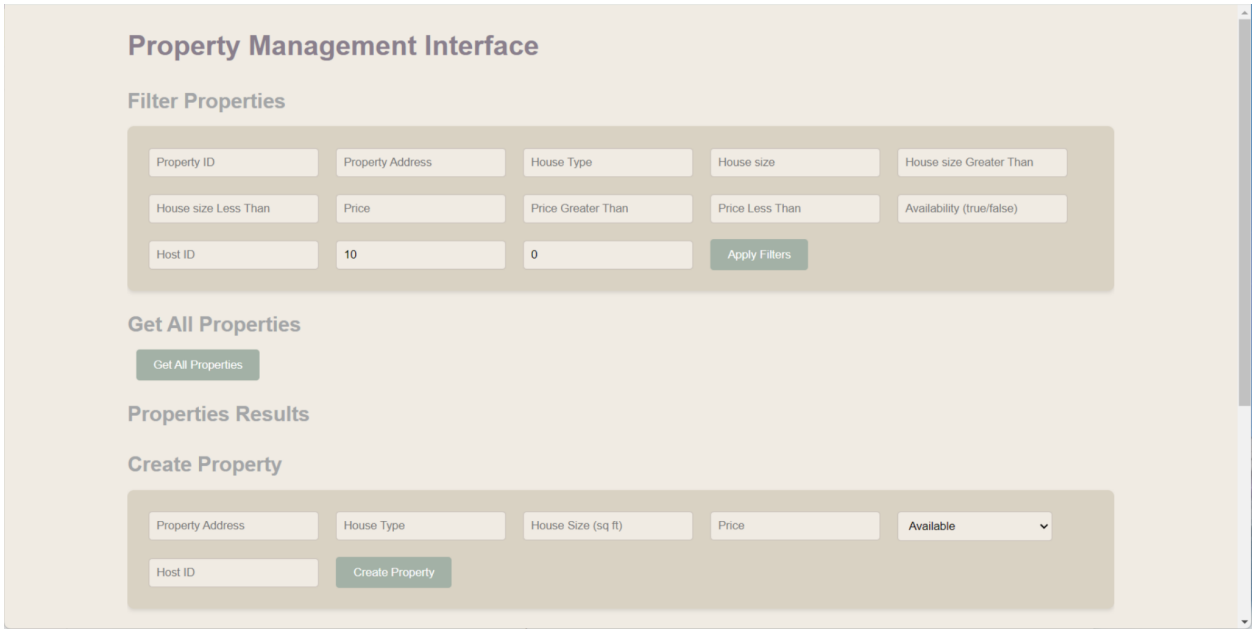
User Management (EC2 with Docker):
http://ec2-3-144-182-9.us-east-2.compute.amazonaws.com:8012

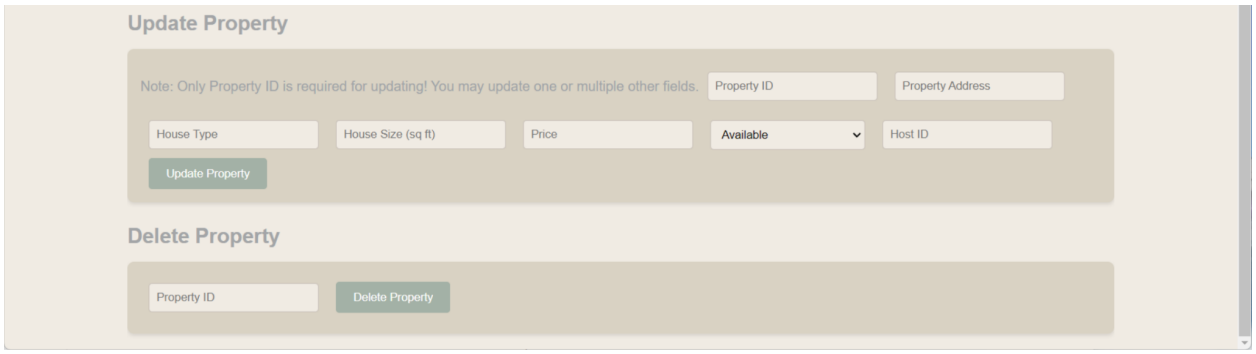


Booking & Review Service (EC2):
http://ec2-3-144-93-114.us-east-2.compute.amazonaws.com:8012



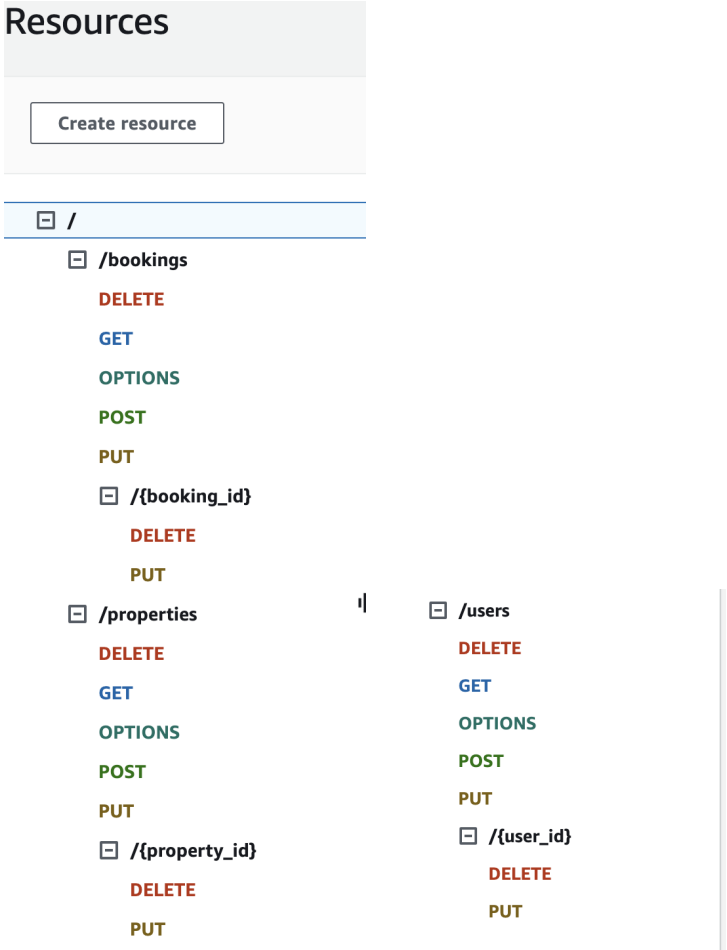
Property Management Service on GCP:
https://e6156-i-am-bezos-402423.ue.r.appspot.com





Note: frontend implementation is not restricted to the above screenshots, please refer to our demo for more connected frontend implementations

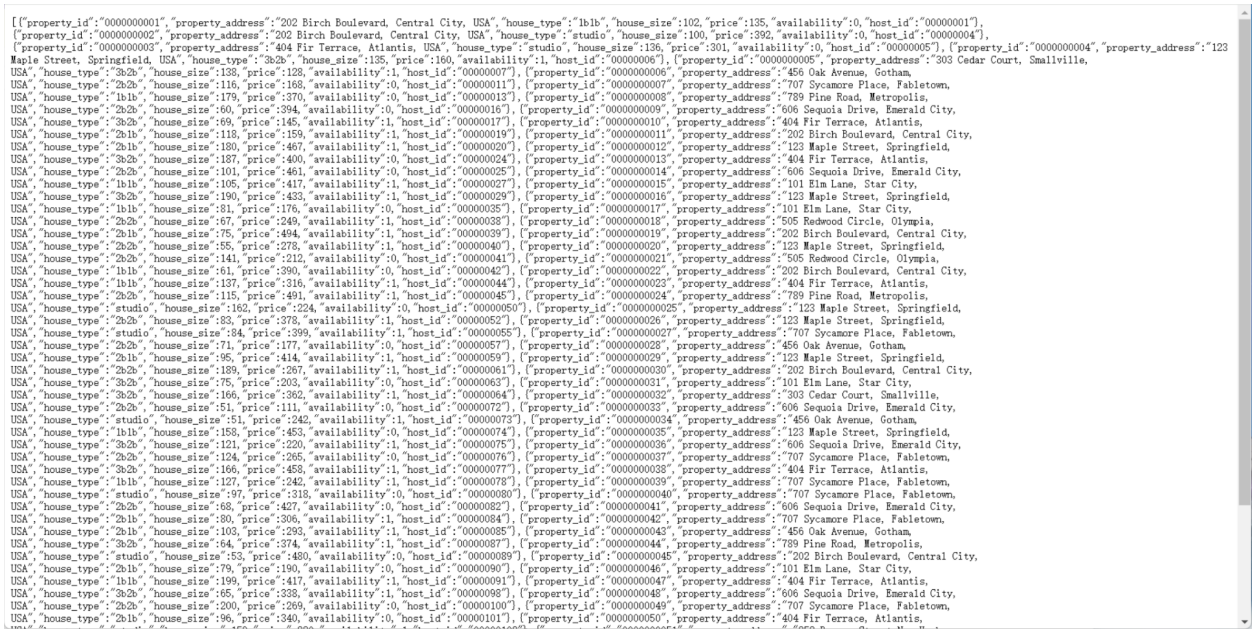
API gateway:
https://eqfosdyv30.execute-api.us-east-1.amazonaws.com/v1/



(with resources):
https://eqfosdyv30.execute-api.us-east-1.amazonaws.com/v1/users (unauthorized if directly click, due to authorizer & jwt token implementation)

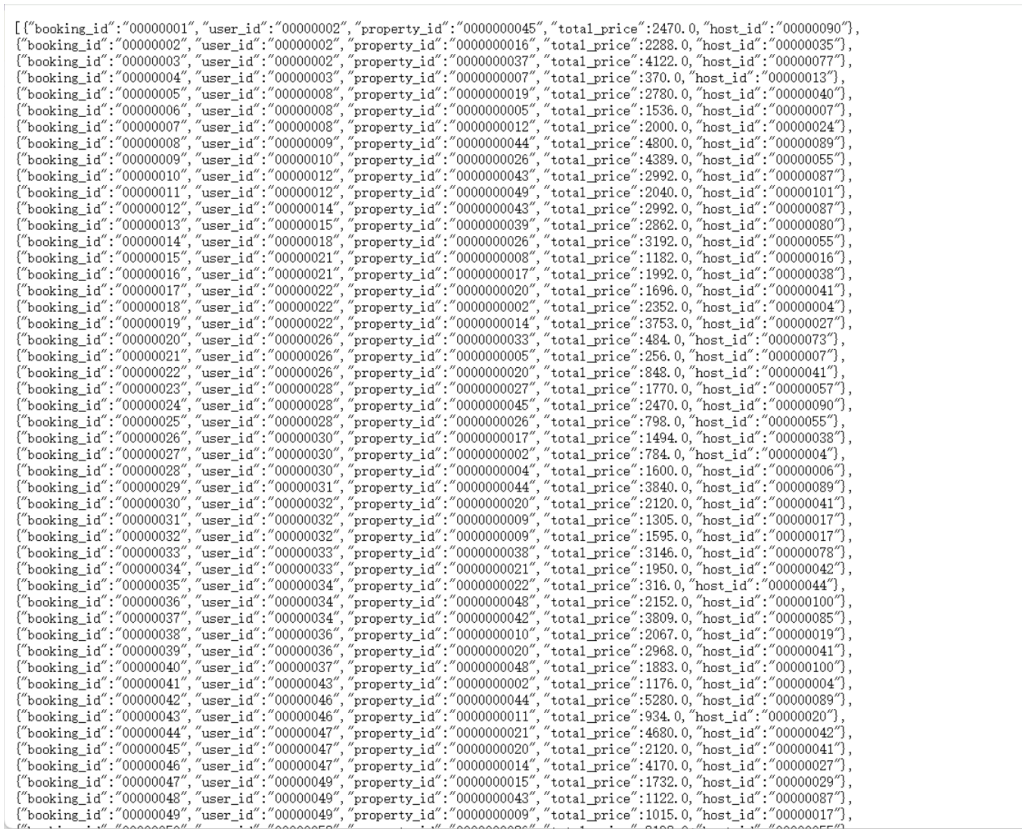
{"message":"Missing Authentication Token"}

https://eqfosdyv30.execute-api.us-east-1.amazonaws.com/v1/properties



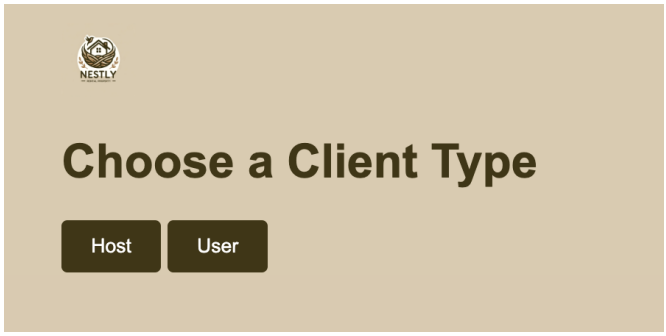
[{"property_id":"0000000001","property_address":"202 Birch Boulevard, Central City, USA","house_type":"1b1b","house_size":102,"price":135,"availability":0,"host_id":"00000001"},
{"property_id":"0000000002","property_address":"202 Birch Boulevard, Central City, USA","house_type":"studio","house_size":100,"price":392,"availability":0,"host_id":"00000004"},
{"property_id":"0000000003","property_address":"404 Fir Terrace, Atlantis, USA","house_type":"studio","house_size":136,"price":301,"availability":0,"host_id":"00000005"},{"property_id":"0000000004","property_address":"123 Maple Street, Springfield, USA","house_type":"3b2b","house_size":135,"price":160,"availability":1,"host_id":"00000006"},{"property_id":"0000000005","property_address":"303 Cedar Court, Smallville, USA","house_type":"3b2b","house_size":138,"price":128,"availability":1,"host_id":"00000007"},{"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":"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":"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"},{"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":"0000000016","property_address":"123 Maple Street, Springfield, USA","house_type":"1b1b","house_size":81,"price":176,"availability":0,"host_id":"00000035"},{"property_id":"0000000017","property_address":"101 Elm Lane, Star City, USA","house_type":"2b2b","house_size":67,"price":249,"availability":1,"host_id":"00000038"},{"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":"0000000019","property_address":"202 Birch Boulevard, Central City, USA","house_type":"2b2b","house_size":55,"price":278,"availability":1,"host_id":"00000040"},{"property_id":"0000000020","property_address":"123 Maple Street, Springfield, USA","house_type":"2b2b","house_size":141,"price":212,"availability":0,"host_id":"00000041"},{"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":"0000000022","property_address":"202 Birch Boulevard, Central City, USA","house_type":"1b1b","house_size":137,"price":316,"availability":1,"host_id":"00000044"},{"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":"0000000024","property_address":"789 Pine Road, Metropolis, USA","house_type":"studio","house_size":162,"price":224,"availability":0,"host_id":"00000050"},{"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":"0000000027","property_address":"707 Sycamore Place, Fabletown, USA","house_type":"2b2b","house_size":71,"price":177,"availability":0,"host_id":"00000057"},{"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":"0000000029","property_address":"123 Maple Street, Springfield, USA","house_type":"2b2b","house_size":189,"price":267,"availability":1,"host_id":"00000061"},{"property_id":"0000000030","property_address":"202 Birch Boulevard, Central City, USA","house_type":"3b2b","house_size":75,"price":203,"availability":0,"host_id":"00000063"},{"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":"0000000032","property_address":"303 Cedar Court, Smallville, USA","house_type":"2b2b","house_size":51,"price":111,"availability":0,"host_id":"00000072"},{"property_id":"0000000033","property_address":"606 Sequoia Drive, Emerald City, USA","house_type":"studio","house_size":51,"price":242,"availability":1,"host_id":"00000073"},{"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":"0000000035","property_address":"123 Maple Street, Springfield, USA","house_type":"3b2b","house_size":121,"price":220,"availability":1,"host_id":"00000075"},{"property_id":"0000000036","property_address":"606 Sequoia Drive, Emerald City, USA","house_type":"2b2b","house_size":124,"price":265,"availability":0,"host_id":"00000076"},{"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":"0000000038","property_address":"404 Fir Terrace, Atlantis, USA","house_type":"1b1b","house_size":127,"price":242,"availability":1,"host_id":"00000078"},{"property_id":"0000000039","property_address":"707 Sycamore Place, Fabletown, USA","house_type":"studio","house_size":97,"price":318,"availability":0,"host_id":"00000080"},{"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":"0000000041","property_address":"606 Sequoia Drive, Emerald City, USA","house_type":"2b1b","house_size":80,"price":306,"availability":1,"host_id":"00000084"},{"property_id":"0000000042","property_address":"707 Sycamore Place, Fabletown, USA","house_type":"2b1b","house_size":103,"price":293,"availability":1,"host_id":"00000085"},{"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":"0000000045","property_address":"202 Birch Boulevard, Central City, USA","house_type":"2b1b","house_size":79,"price":190,"availability":0,"host_id":"00000090"},{"property_id":"0000000046","property_address":"101 Elm Lane, Star City, USA","house_type":"1b1b","house_size":199,"price":417,"availability":1,"host_id":"00000091"},{"property_id":"0000000047","property_address":"404 Fir Terrace, Atlantis, USA","house_type":"3b2b","house_size":65,"price":338,"availability":1,"host_id":"00000098"},{"property_id":"0000000048","property_address":"606 Sequoia Drive, Emerald City, USA","house_type":"2b2b","house_size":200,"price":269,"availability":0,"host_id":"00000100"},{"property_id":"0000000049","property_address":"707 Sycamore Place, Fabletown, USA","house_type":"2b1b","house_size":96,"price":340,"availability":0,"host_id":"00000101"},{"property_id":"0000000050","property_address":"404 Fir Terrace, Atlantis,

https://eqfosdyv30.execute-api.us-east-1.amazonaws.com/v1/bookings)

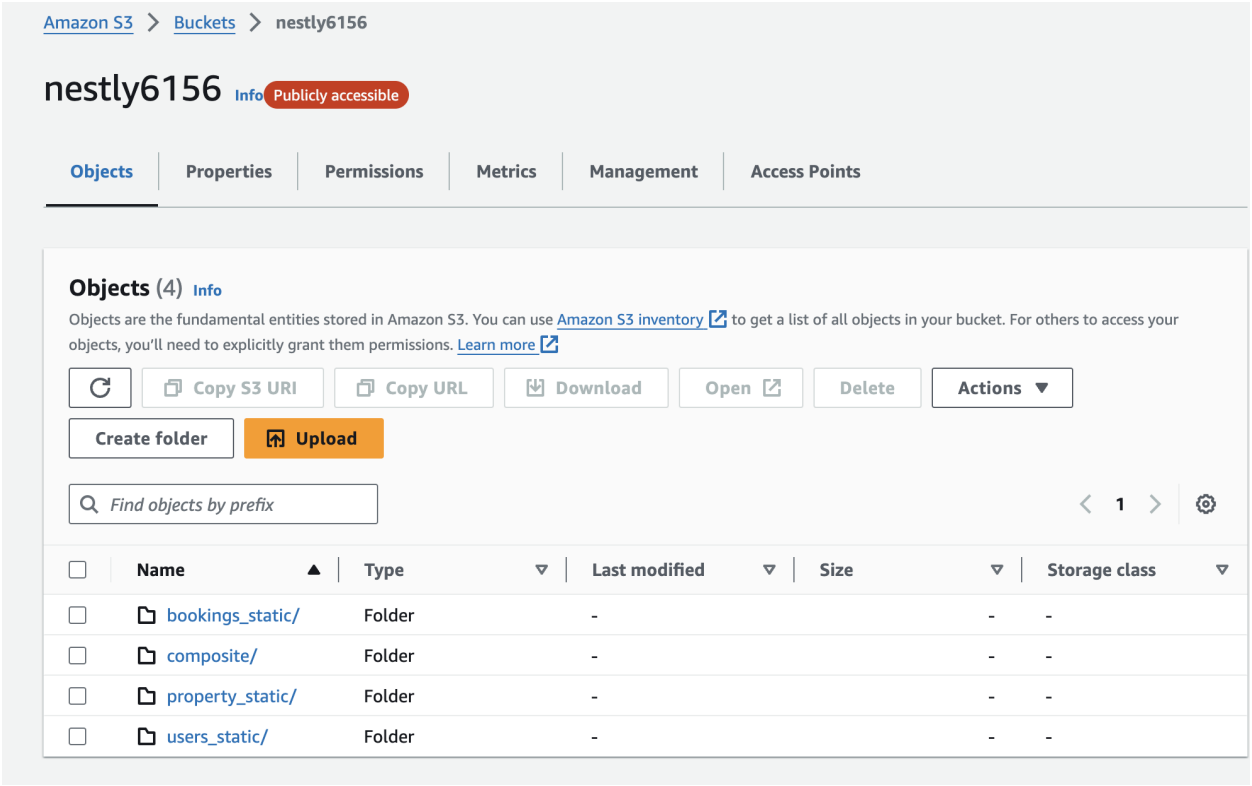


[{"booking_id":"00000001","user_id":"00000002","property_id":"0000000045","total_price":2470.0,"host_id":"00000090"},
{"booking_id":"00000002","user_id":"00000002","property_id":"0000000016","total_price":2288.0,"host_id":"00000035"},
{"booking_id":"00000003","user_id":"00000002","property_id":"0000000037","total_price":4122.0,"host_id":"00000077"},
{"booking_id":"00000004","user_id":"00000003","property_id":"0000000007","total_price":370.0,"host_id":"00000013"},
{"booking_id":"00000005","user_id":"00000008","property_id":"0000000019","total_price":2780.0,"host_id":"00000040"},
{"booking_id":"00000006","user_id":"00000008","property_id":"0000000005","total_price":1536.0,"host_id":"00000007"},
{"booking_id":"00000007","user_id":"00000008","property_id":"0000000012","total_price":2000.0,"host_id":"00000024"},
{"booking_id":"00000008","user_id":"00000009","property_id":"0000000044","total_price":4800.0,"host_id":"00000089"},
{"booking_id":"00000009","user_id":"00000010","property_id":"0000000026","total_price":4389.0,"host_id":"00000055"},
{"booking_id":"00000010","user_id":"00000012","property_id":"0000000043","total_price":2992.0,"host_id":"00000087"},
{"booking_id":"00000011","user_id":"00000012","property_id":"0000000049","total_price":2040.0,"host_id":"00000101"},
{"booking_id":"00000012","user_id":"00000014","property_id":"0000000043","total_price":2992.0,"host_id":"00000087"},
{"booking_id":"00000013","user_id":"00000015","property_id":"0000000039","total_price":2862.0,"host_id":"00000080"},
{"booking_id":"00000014","user_id":"00000018","property_id":"0000000026","total_price":3192.0,"host_id":"00000055"},
{"booking_id":"00000015","user_id":"00000021","property_id":"0000000008","total_price":1182.0,"host_id":"00000016"},
{"booking_id":"00000016","user_id":"00000021","property_id":"0000000017","total_price":1992.0,"host_id":"00000038"},
{"booking_id":"00000017","user_id":"00000022","property_id":"0000000020","total_price":1696.0,"host_id":"00000041"},
{"booking_id":"00000018","user_id":"00000022","property_id":"0000000002","total_price":2352.0,"host_id":"00000004"},
{"booking_id":"00000019","user_id":"00000022","property_id":"0000000014","total_price":3753.0,"host_id":"00000027"},
{"booking_id":"00000020","user_id":"00000026","property_id":"0000000033","total_price":484.0,"host_id":"00000073"},
{"booking_id":"00000021","user_id":"00000026","property_id":"0000000005","total_price":256.0,"host_id":"00000007"},
{"booking_id":"00000022","user_id":"00000026","property_id":"0000000020","total_price":848.0,"host_id":"00000041"},
{"booking_id":"00000023","user_id":"00000028","property_id":"0000000027","total_price":1770.0,"host_id":"00000057"},
{"booking_id":"00000024","user_id":"00000028","property_id":"0000000045","total_price":2470.0,"host_id":"00000090"},
{"booking_id":"00000025","user_id":"00000028","property_id":"0000000026","total_price":798.0,"host_id":"00000055"},
{"booking_id":"00000026","user_id":"00000030","property_id":"0000000017","total_price":1494.0,"host_id":"00000038"},
{"booking_id":"00000027","user_id":"00000030","property_id":"0000000002","total_price":784.0,"host_id":"00000004"},
{"booking_id":"00000028","user_id":"00000030","property_id":"0000000004","total_price":1600.0,"host_id":"00000006"},
{"booking_id":"00000029","user_id":"00000031","property_id":"0000000044","total_price":3840.0,"host_id":"00000089"},
{"booking_id":"00000030","user_id":"00000032","property_id":"0000000020","total_price":2120.0,"host_id":"00000041"},
{"booking_id":"00000031","user_id":"00000032","property_id":"0000000009","total_price":1305.0,"host_id":"00000017"},
{"booking_id":"00000032","user_id":"00000032","property_id":"0000000009","total_price":1595.0,"host_id":"00000017"},
{"booking_id":"00000033","user_id":"00000033","property_id":"0000000038","total_price":3146.0,"host_id":"00000078"},
{"booking_id":"00000034","user_id":"00000033","property_id":"0000000021","total_price":1950.0,"host_id":"00000042"},
{"booking_id":"00000035","user_id":"00000034","property_id":"0000000022","total_price":316.0,"host_id":"00000044"},
{"booking_id":"00000036","user_id":"00000034","property_id":"0000000048","total_price":2152.0,"host_id":"00000100"},
{"booking_id":"00000037","user_id":"00000034","property_id":"0000000042","total_price":3809.0,"host_id":"00000085"},
{"booking_id":"00000038","user_id":"00000036","property_id":"0000000010","total_price":2067.0,"host_id":"00000019"},
{"booking_id":"00000039","user_id":"00000036","property_id":"0000000020","total_price":2968.0,"host_id":"00000041"},
{"booking_id":"00000040","user_id":"00000037","property_id":"0000000048","total_price":1883.0,"host_id":"00000100"},
{"booking_id":"00000041","user_id":"00000043","property_id":"0000000002","total_price":1176.0,"host_id":"00000004"},
{"booking_id":"00000042","user_id":"00000046","property_id":"0000000044","total_price":5280.0,"host_id":"00000089"},
{"booking_id":"00000043","user_id":"00000046","property_id":"0000000011","total_price":934.0,"host_id":"00000020"},
{"booking_id":"00000044","user_id":"00000047","property_id":"0000000021","total_price":4680.0,"host_id":"00000042"},
{"booking_id":"00000045","user_id":"00000047","property_id":"0000000020","total_price":2120.0,"host_id":"00000041"},
{"booking_id":"00000046","user_id":"00000047","property_id":"0000000014","total_price":4170.0,"host_id":"00000027"},
{"booking_id":"00000047","user_id":"00000049","property_id":"0000000015","total_price":1732.0,"host_id":"00000029"},
{"booking_id":"00000048","user_id":"00000049","property_id":"0000000043","total_price":1122.0,"host_id":"00000087"},
{"booking_id":"00000049","user_id":"00000049","property_id":"0000000009","total_price":1015.0,"host_id":"00000017"},

Composite:
http://ec2-52-14-240-60.us-east-2.compute.amazonaws.com:8012/



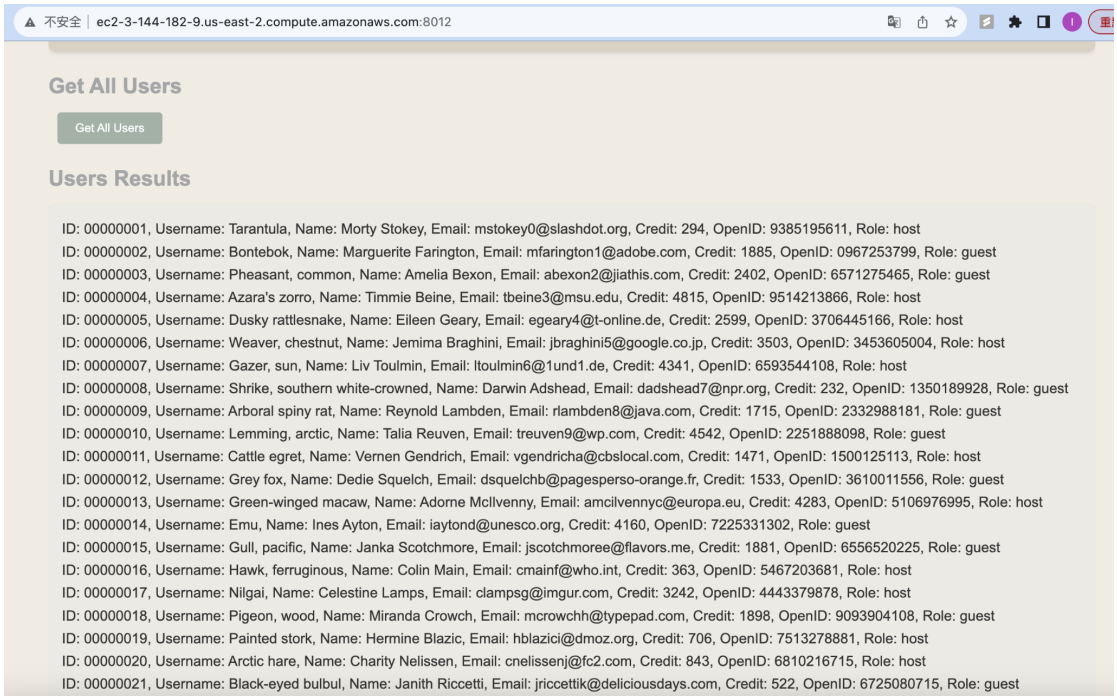
S3:
https://nestly6156.s3.us-east-2.amazonaws.com



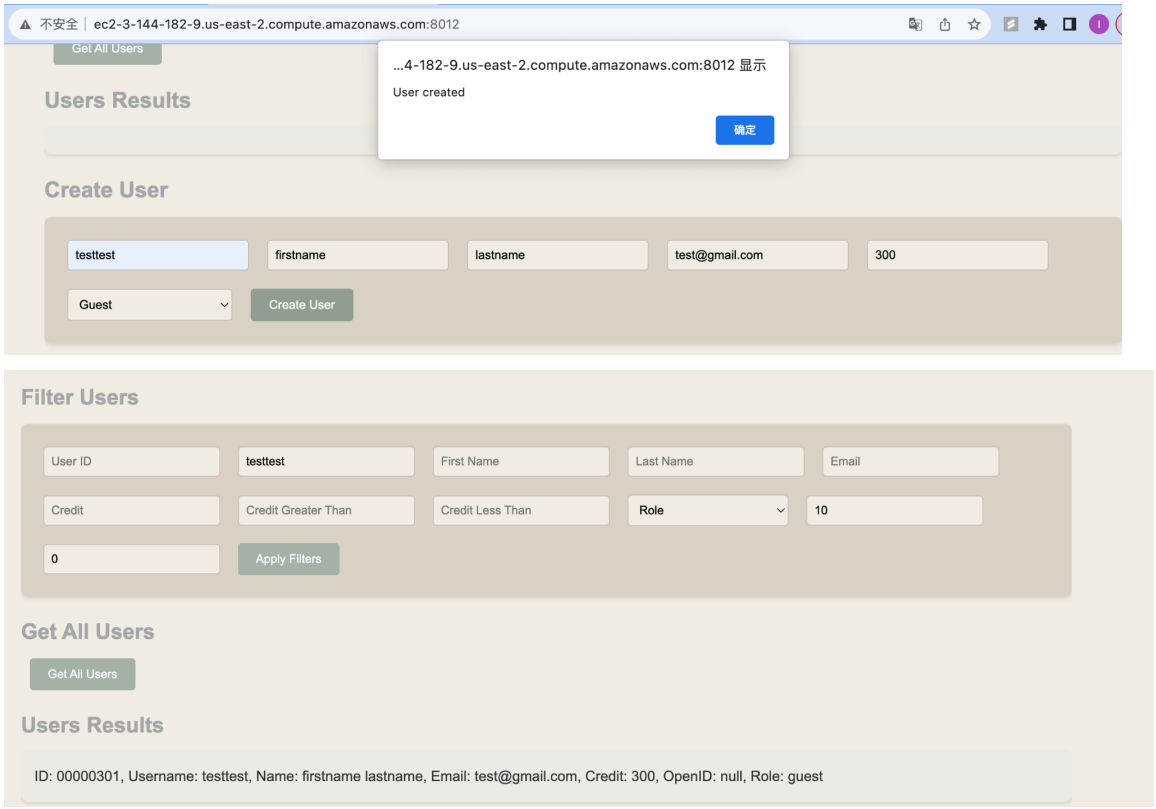
Implementation Screenshots

Enhance Microservices Implementation:

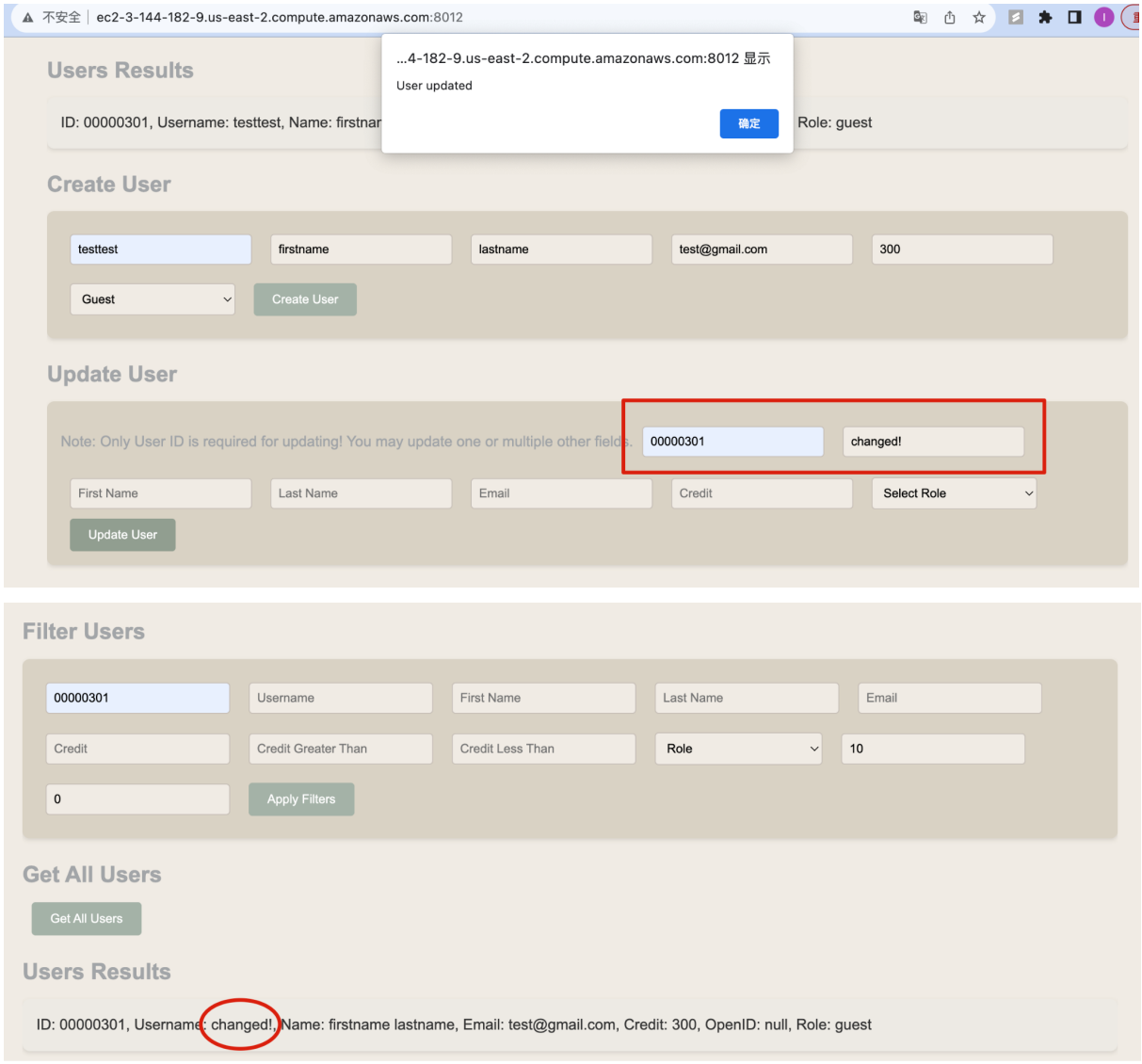
CRUD (finished on all 3 microservices, using Users as example):
GET: (getting all here)



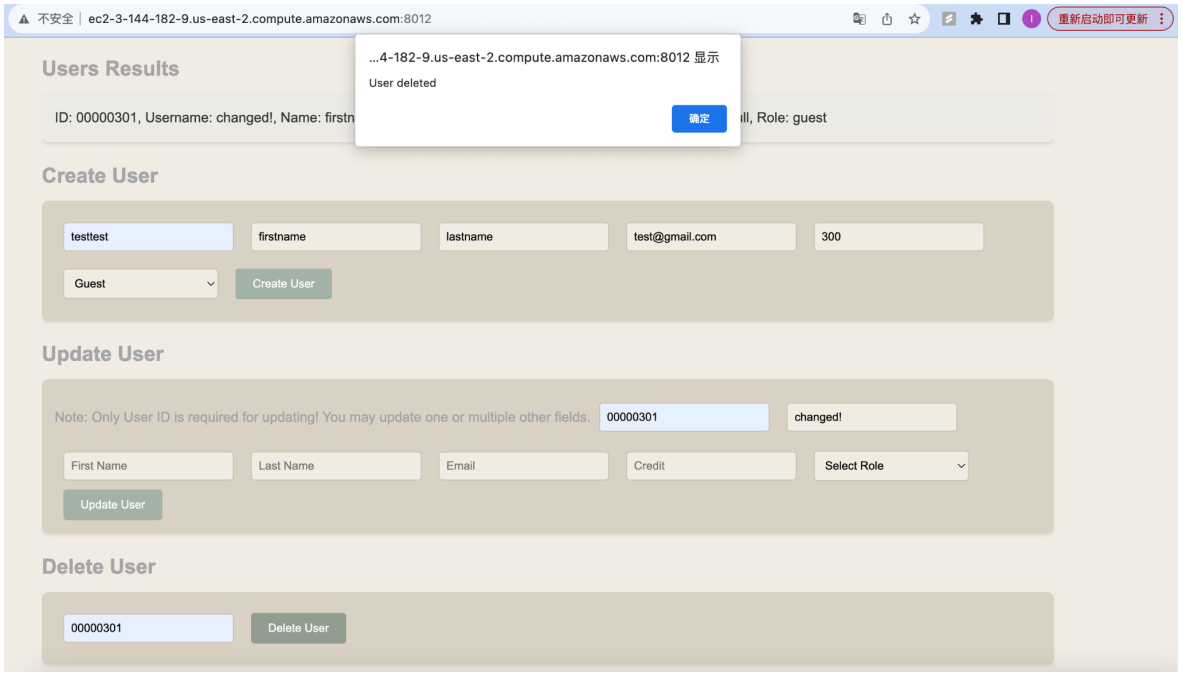
PUT: (added a test user ‘testtest’ and the GET on username=testtest)

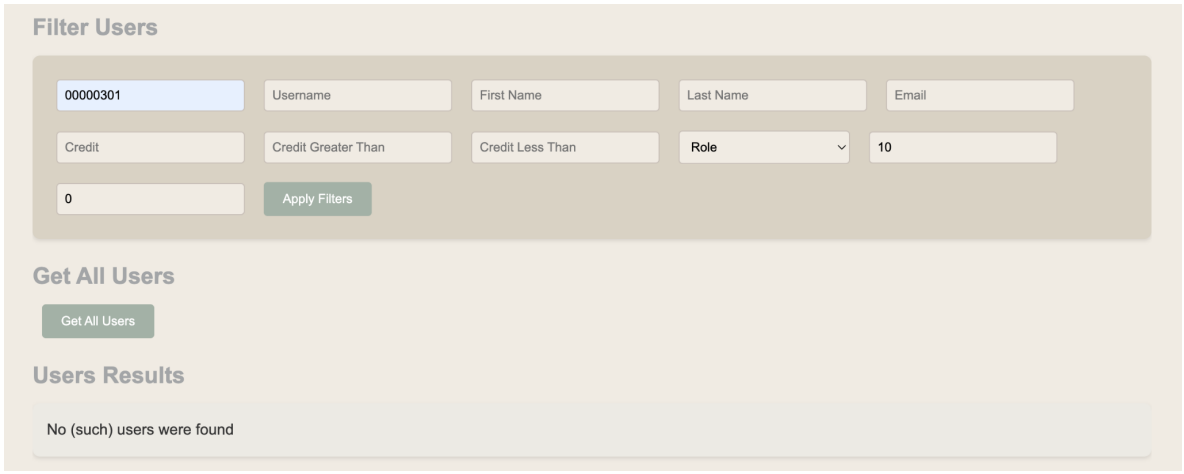


POST: (updated the added user using user id=00000301, username to ‘changed!’, and the GET on user id=00000301, showing username changed to ‘changed!’)

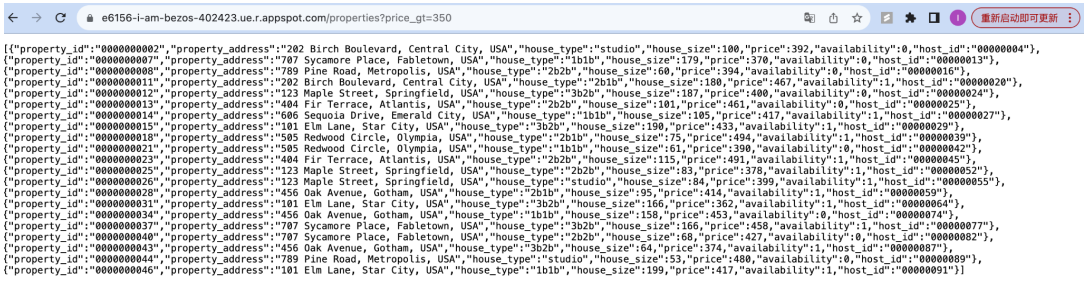


DELETE (DELETE user id=00000301, and GET user id=00000301 returning “No (such) users were found”):



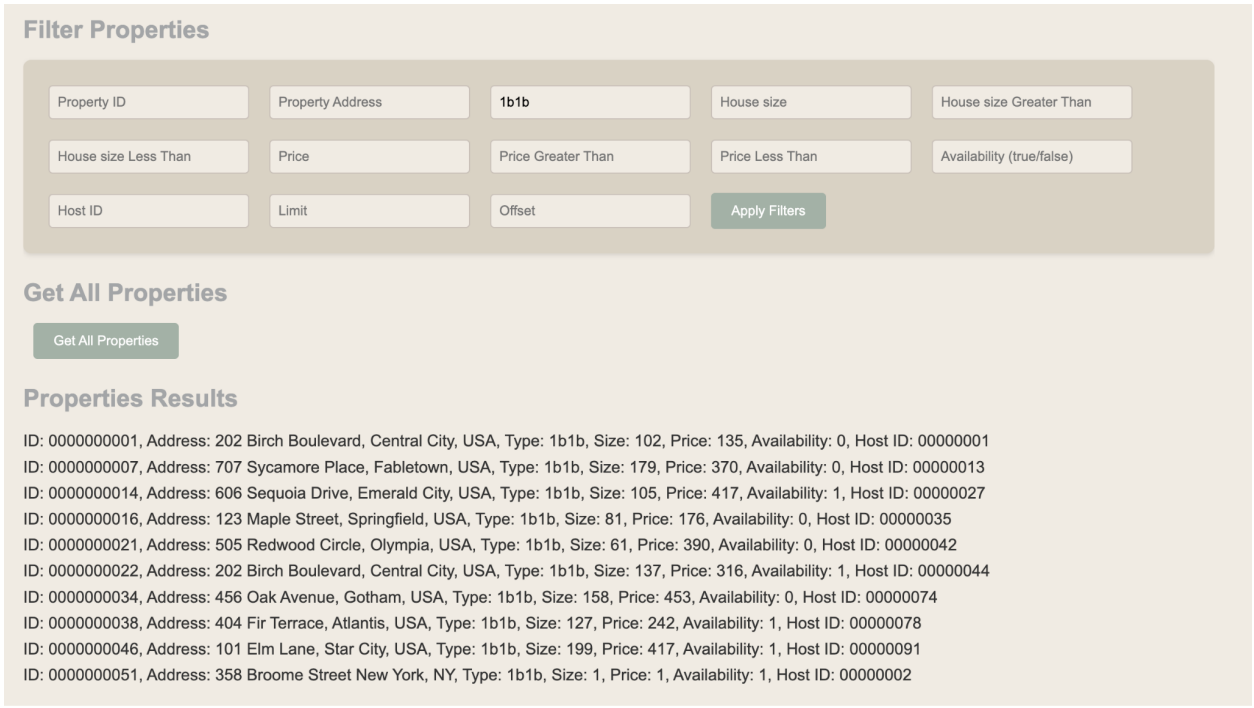


Querying (realized on Properties)
(E.g. price_gt=350, return properties with price greater than 350)



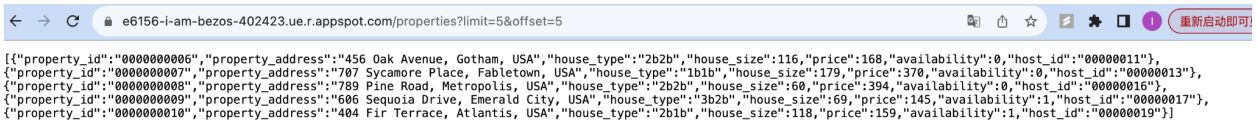
[{"property_id":"0000000002","property_address":"202 Birch Boulevard, Central City, USA","house_type":"studio","house_size":100,"price":392,"availability":0,"host_id":"00000004"},
{"property_id":"0000000007","property_address":"707 Sycamore Place, Fabletown, USA","house_type":"1b1b","house_size":179,"price":370,"availability":0,"host_id":"00000013"},
{"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"}]

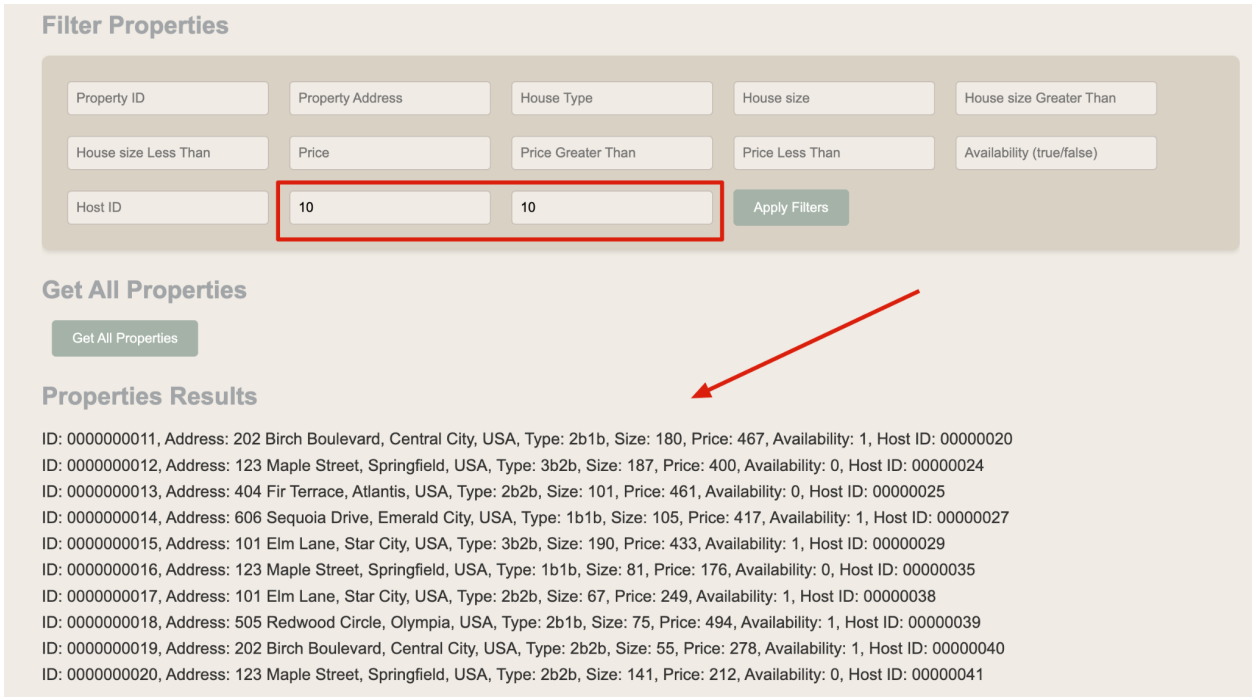
(E.g. with front end, house_type = 1b1b)



Pagination & offset (realized on Properties)

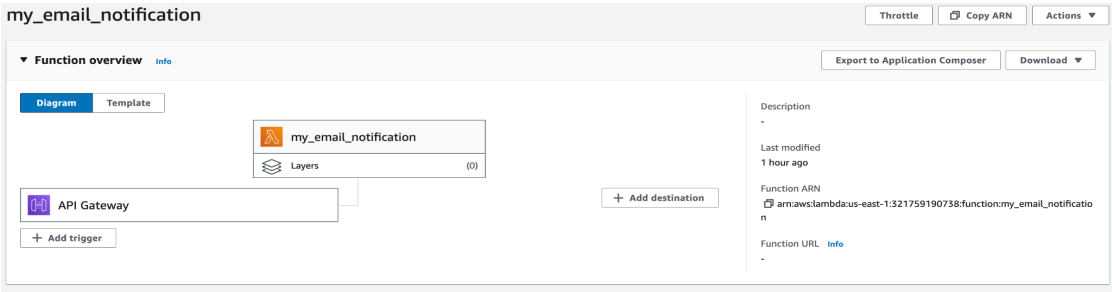
(E.g. Limit = 5 & Offset = 5 -> we got IDs from 6 to 10, instead of from 1 to 50)

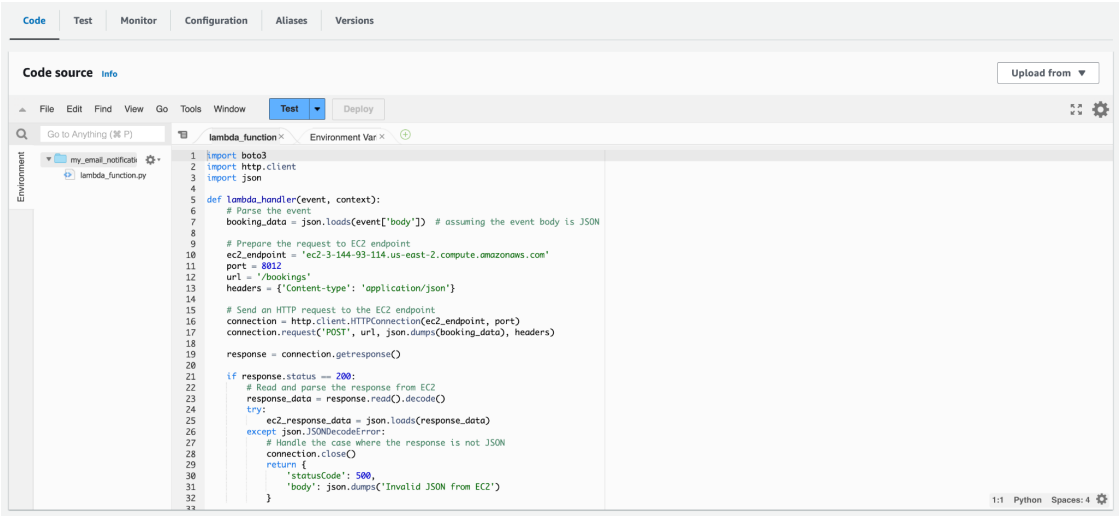




Events, Notifications, Pub/Sub:

We implemented a serverless architecture to handle event-driven notifications for new bookings. The system utilizes AWS Lambda, triggered by the API Gateway, to process booking events. Upon a new booking creation, the Lambda function parses the booking details and sends a POST request to an EC2 microservice endpoint to handle the booking logic. After receiving a successful response from the EC2 service, the Lambda function publishes a message to an Amazon Simple Notification Service (SNS) topic, which is configured to send an email notification to the host. This seamless integration of services ensures that the host is promptly notified of new bookings, enhancing the efficiency of the booking system.

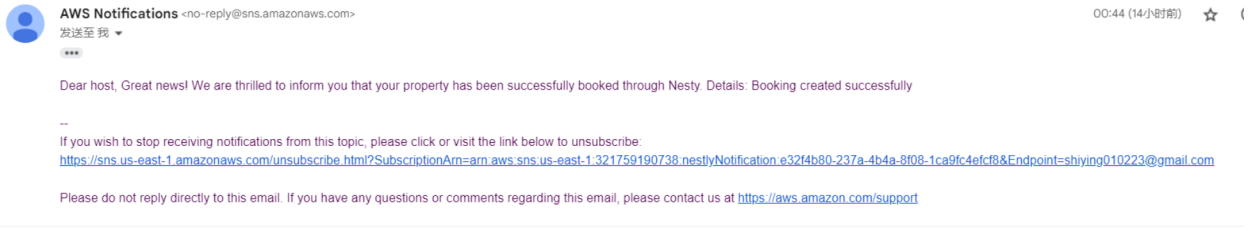




```
    # Send an email (using SNS or SES)
    sns_client = boto3.client('sns')
    snsArn = 'arn:aws:sns:us-east-1:321759190738:nestlyNotification'
    message = f"Dear host, Great news! We are thrilled to inform you that your property has been successfully booked through Nesty. Details: {ec2_response_data}"

    sns_response = sns_client.publish(
        TopicArn=snsArn,
        Message=message,
        Subject='New Booking Alert: Your Property has been Booked on Nesty!'
    )

    connection.close()
    return {
        'statusCode': 200,
        'body': json.dumps('Email sent successfully')
    }
else:
    # Handle errors from EC2 request
    connection.close()
    return {
        'statusCode': response.status,
        'body': json.dumps('Error communicating with EC2')
    }
```



AWS Notifications <no-reply@sns.amazonaws.com> 00:44 (14小时前)

发送至 我

Dear host, Great news! We are thrilled to inform you that your property has been successfully booked through Nesty. Details: Booking created successfully

--

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:321759190738:nestlyNotification:e32f4b80-237a-4b4a-8f08-1ca9fc4efcf8&Endpoint=shiying010223@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

Composition/Aggregators

Sync: (designated order always)

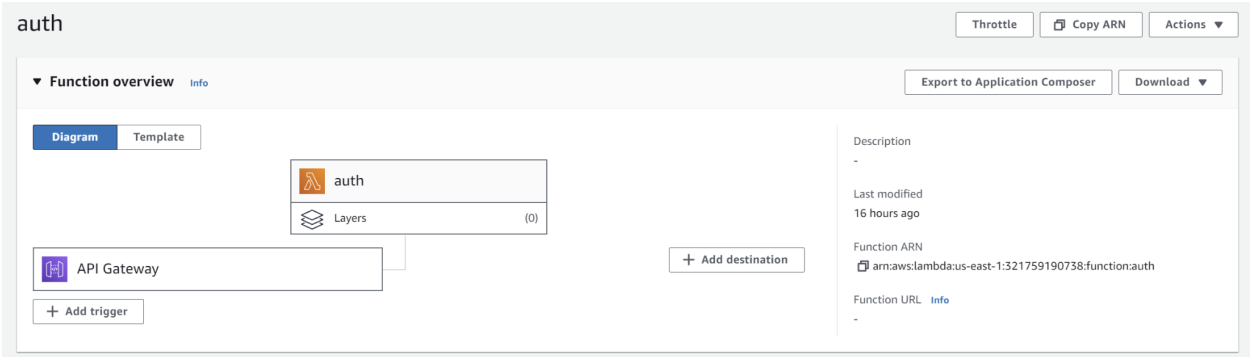
```
INFO:     Started server process [14318]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8012 (Press CTRL+C to quit)
INFO:     127.0.0.1:52368 - "GET /docs HTTP/1.1" 200 OK
INFO:     127.0.0.1:52368 - "GET /openapi.json 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)
```

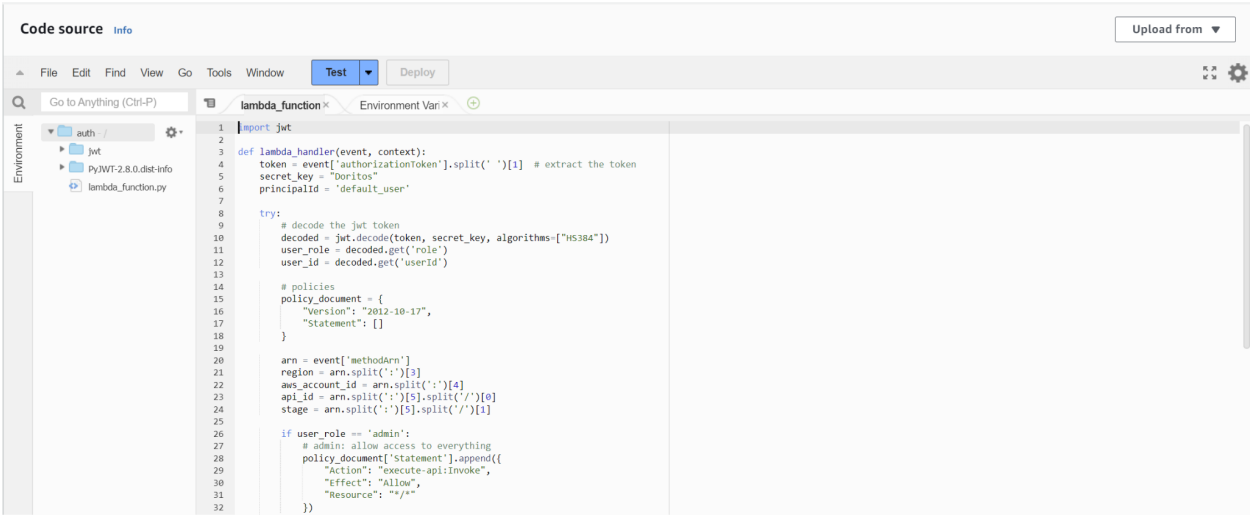
Async: different orders

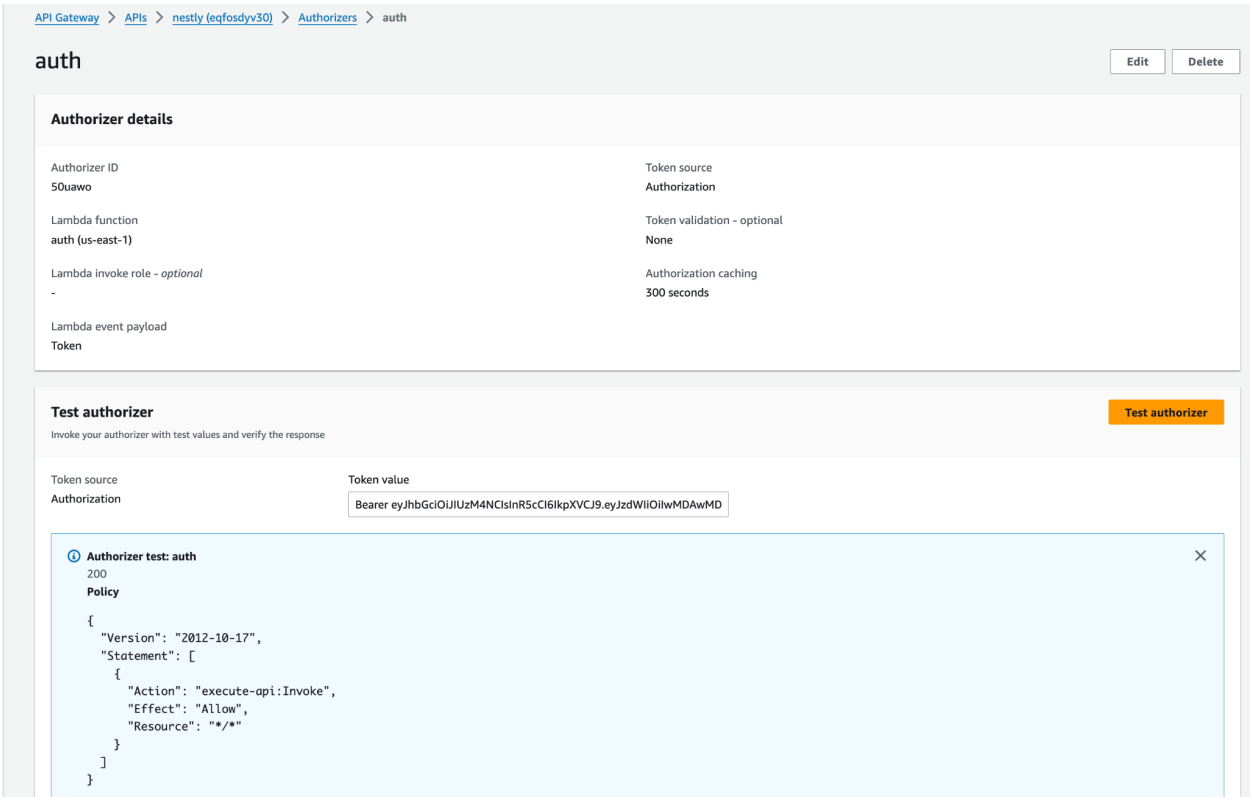
```
INFO:     127.0.0.1:52408 - "GET /async-aggregator HTTP/1.1" 200 OK
users service has returned a response (in async)
properties service has returned a response (in async)
bookings service has returned a response (in async)
INFO:     127.0.0.1:52415 - "GET /async-aggregator HTTP/1.1" 200 OK
bookings 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)
properties service has returned a response (in async)
INFO:     127.0.0.1:52444 - "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:     127.0.0.1:52449 - "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
```

***Authorizer*:**

We implemented a custom Lambda function authorizer that integrates with AWS API Gateway to manage user authentication and authorization through JWT tokens. Upon receiving an API request, the authorizer extracts and decodes the JWT token to verify the user's identity and role. Depending on the role—whether 'admin', 'guest', or 'host'—the function constructs a policy document that outlines the permitted actions and resources for that user. Admin users are granted full access, while guest and host roles have restricted permissions, limited to operations on their respective user information and viewing properties. The authorizer is also equipped to respond to exceptions by issuing messages for expired or invalid tokens, thereby reinforcing the API's security by enforcing role-based access control.



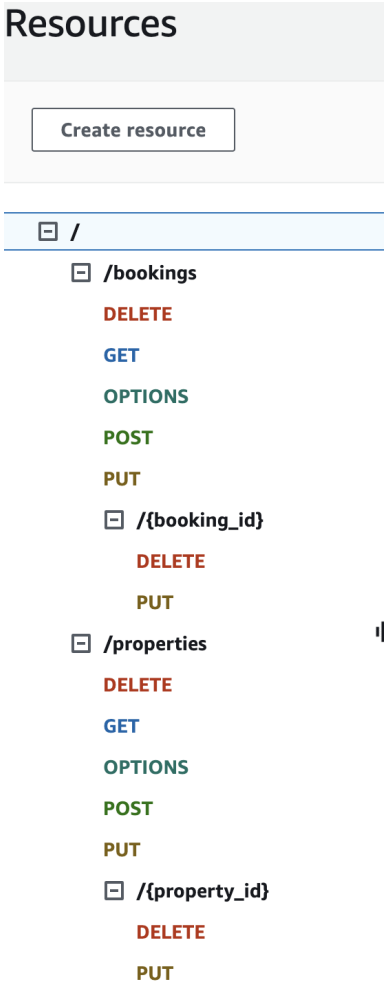




API Gateway:

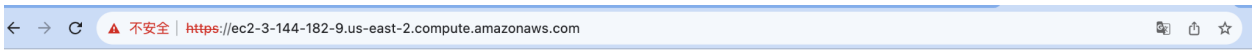
/bookings endpoint allows clients to create a new booking (POST), retrieve a list of all bookings (GET), or delete all bookings (DELETE). It also supports updating booking information (PUT).
/{booking_id} is a parameterized endpoint where clients can target a specific booking by its unique identifier to update (PUT) or delete (DELETE) it.
/properties endpoint functions similarly to /bookings, facilitating the creation (POST), retrieval (GET), deletion (DELETE), and updating (PUT) of property records.

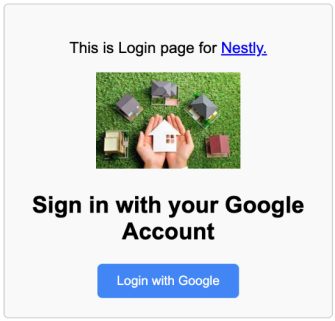
/{property_id} enables specific operations on an individual property identified by its unique ID, allowing for updates (PUT) and deletion (DELETE).
/users endpoint is designed to manage user records, allowing for the creation (POST), retrieval (GET), and deletion (DELETE) of user information, as well as the ability to update user details (PUT) and handle preflight requests for cross-origin resource sharing (OPTIONS).
/{user_id} allows for targeted operations on a user's record by their unique identifier, enabling updates (PUT) and deletion (DELETE).

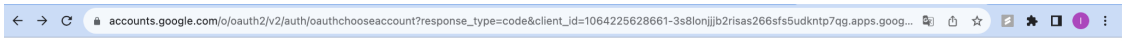


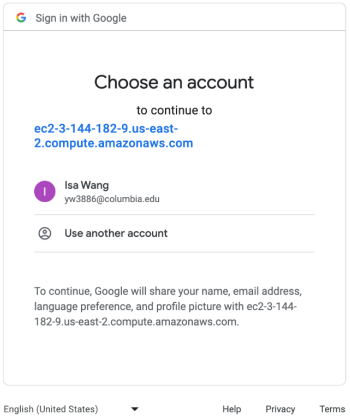
- /users
  - DELETE
  - GET
  - OPTIONS
  - POST
  - PUT
  - /{user_id}
    - DELETE
    - PUT

Single Sign-On (SSO):

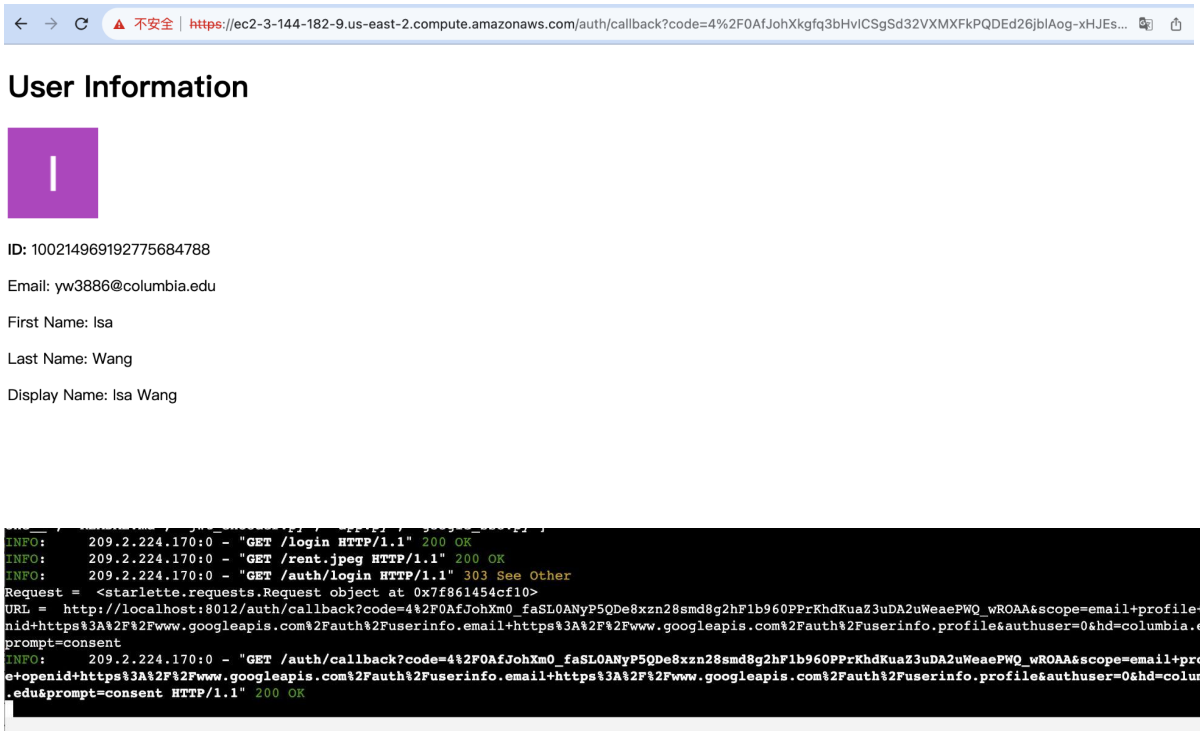






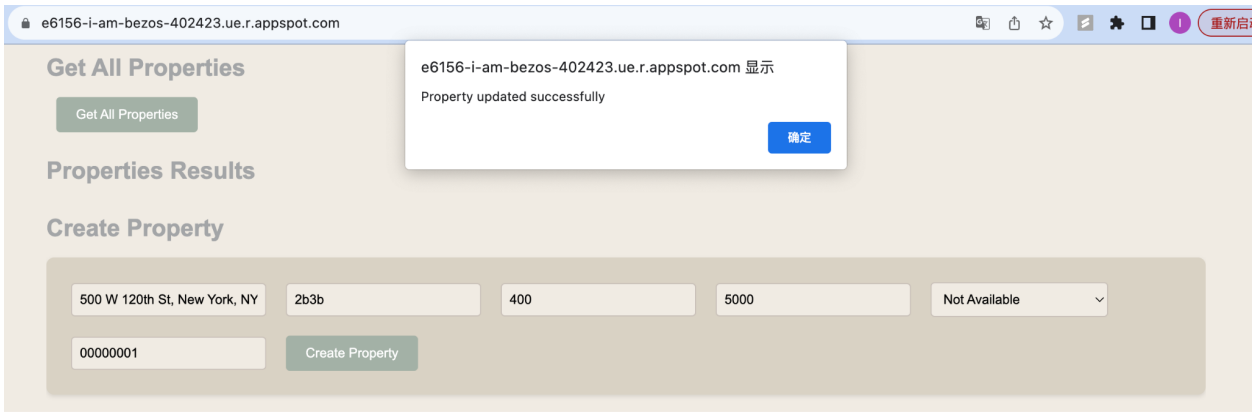


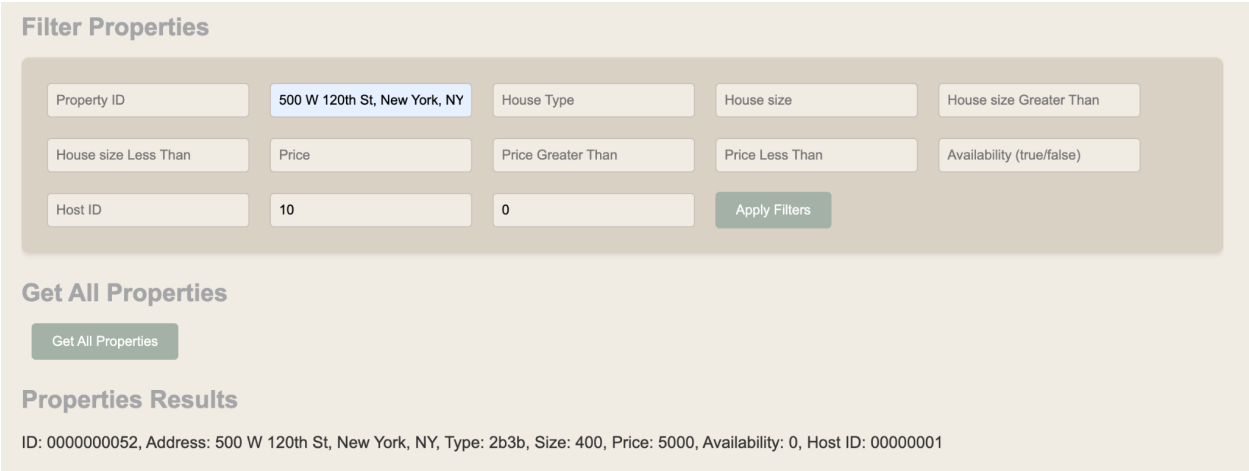
Authorized request to Google:



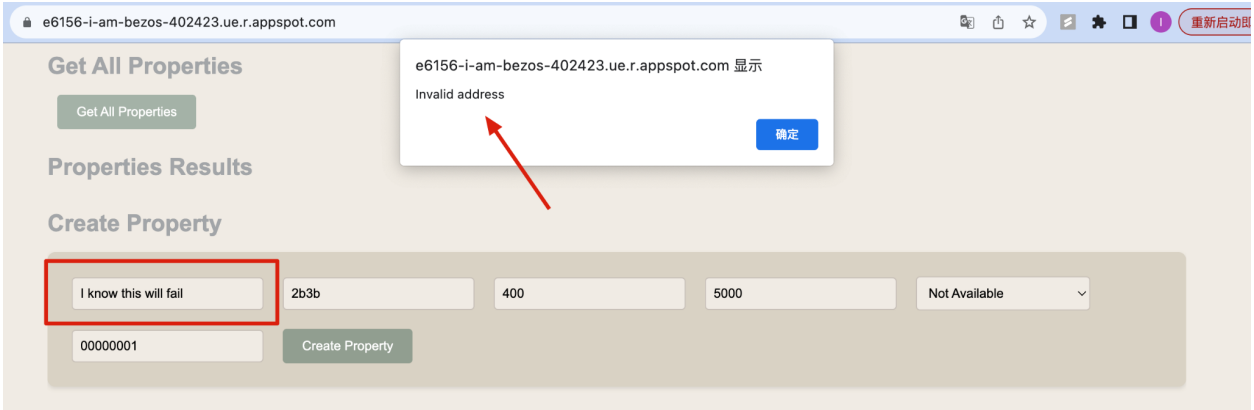
External API:

API: Smarty Street API for Address Validation on Properties microservice.
Functionality: for PUT and POST on Properties, can be successful if address is validated, but returns “Address Invalid” and fails the PUT and POST, if address is not valid.
Success case: (used Valid Address of Mudd Building)

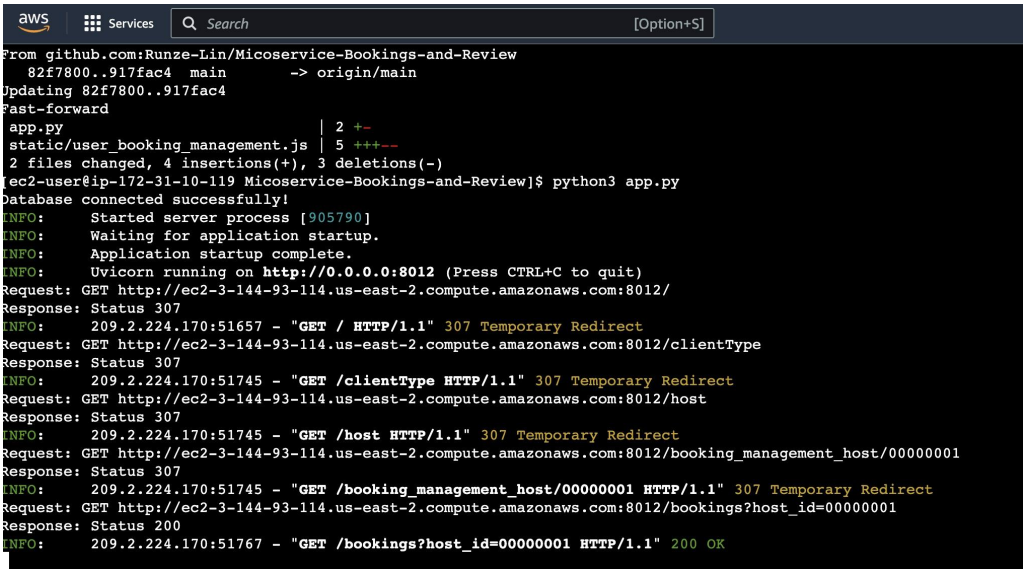




Fail case (use obviously invalid address of “invalid address”):



Middleware

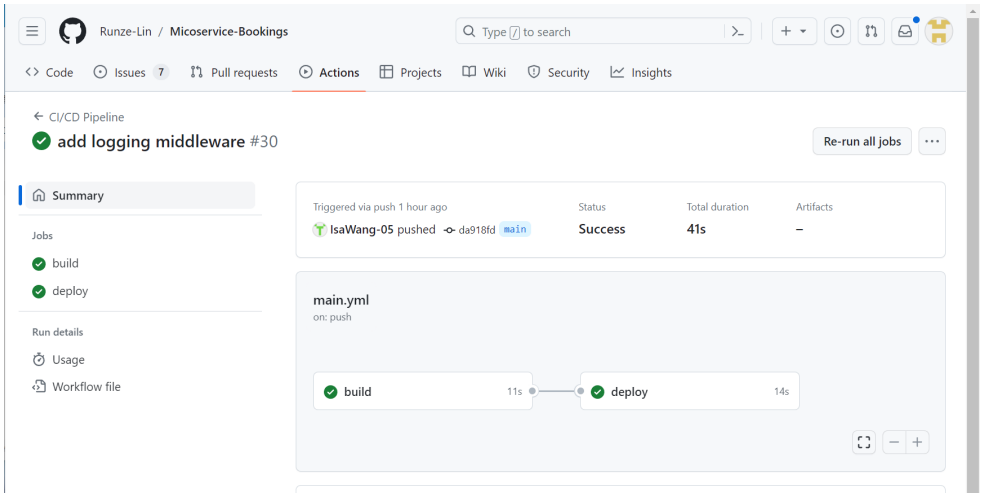


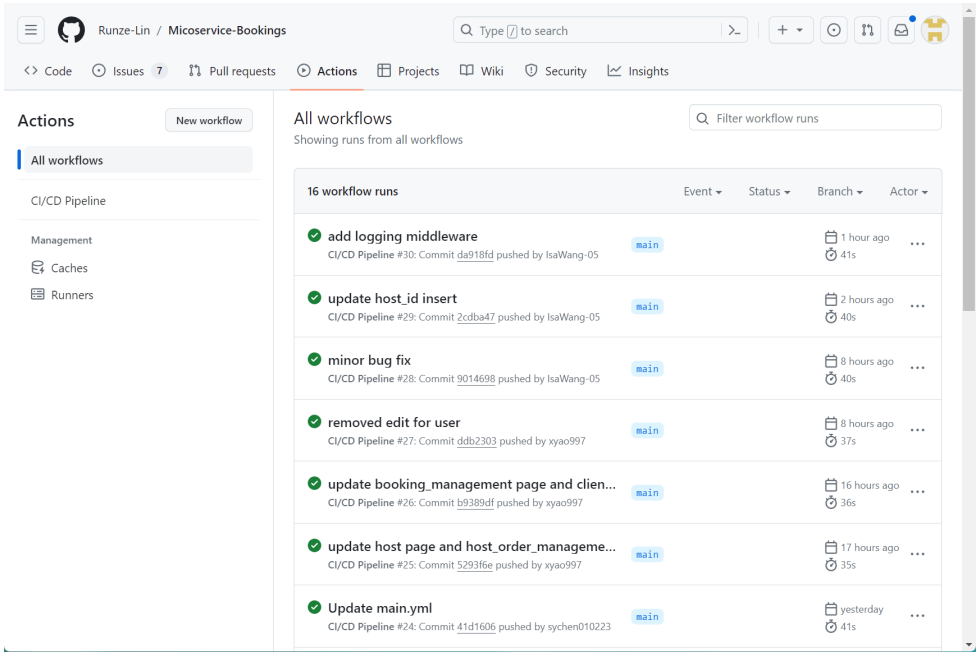
CI/CD

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





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).

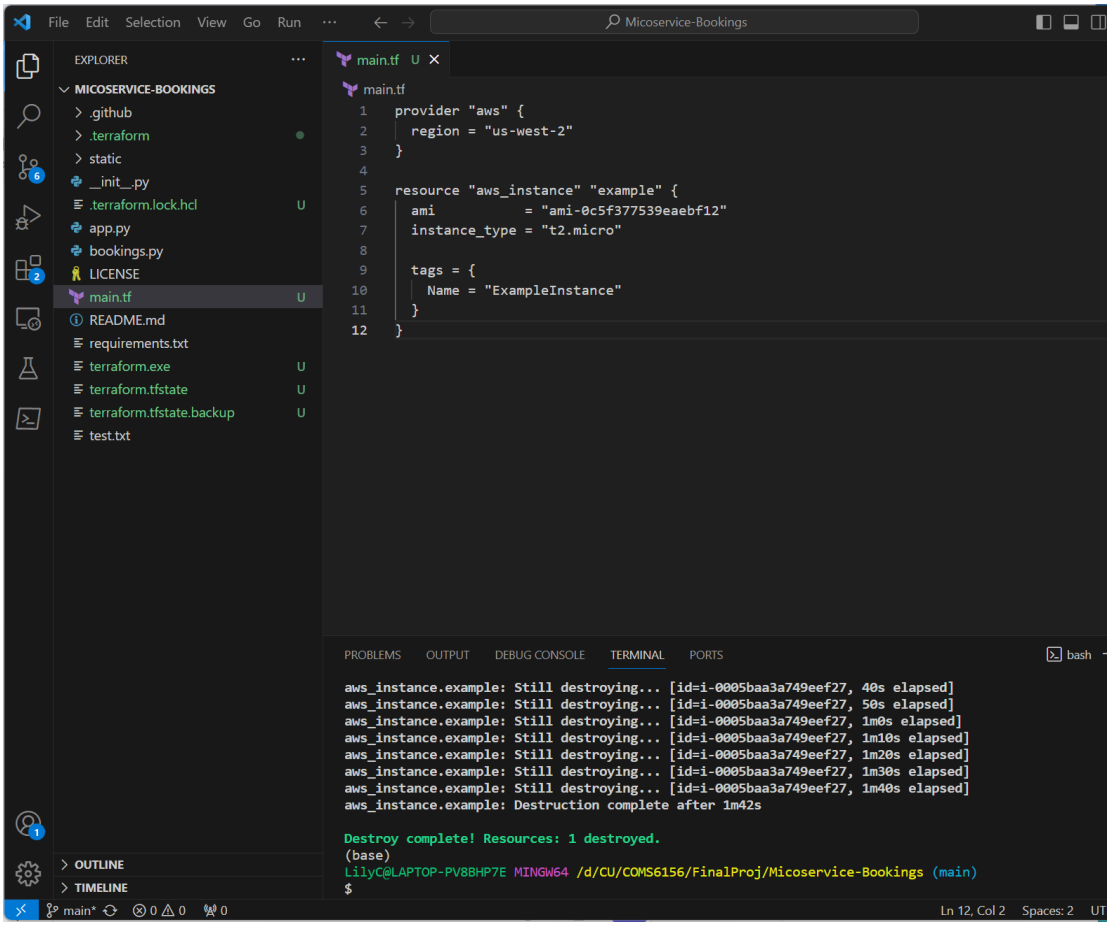
```
Plan: 1 to add, 0 to change, 0 to destroy.

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 (
```



GraphQL

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