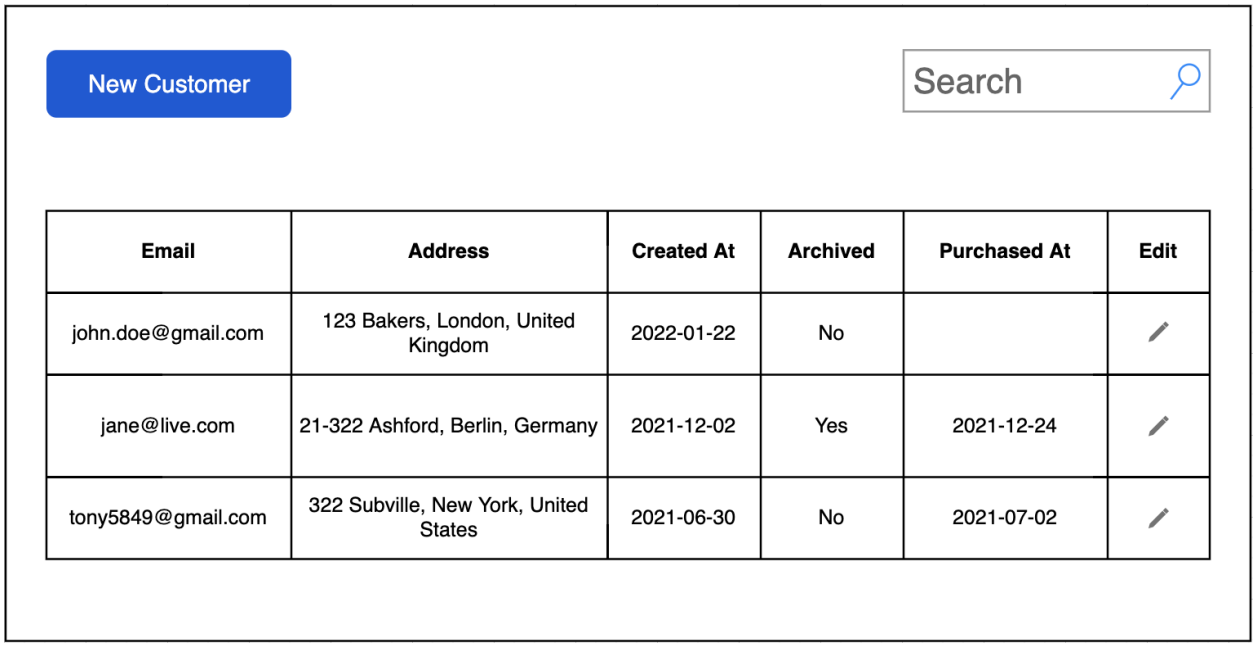
Senior Backend Engineer - .NET

Case study: Customers microservice

The objective is to create a Customers microservice to store a list of customers and notify other services when a certain customer information gets updated.

The Customers microservice is used by e-commerce admins to create, archive and update customers on the admin dashboard.



Email	Address	Created At	Archived	Purchased At	Edit
john.doe@gmail.com	123 Bakers, London, United Kingdom	2022-01-22	No		
jane@live.com	21-322 Ashford, Berlin, Germany	2021-12-02	Yes	2021-12-24	
tony5849@gmail.com	322 Subville, New York, United States	2021-06-30	No	2021-07-02	

Entities

- Customer
  - Email - String, unique
  - Address - Address
  - CreatedAt - UTC DateTime
  - IsArchived - Boolean
  - PurchasedAt - UTC DateTime
- Address
  - Street - String
  - City - String
  - Country - String

All attributes are required.

Integration Events

The Customers microservice is connected to a message broker, where it posts events whenever a customer is created, archived or a customer's address updated. In addition, the microservice is subscribed to the message broker for the OrderCompleted event, which makes the microservice update the “Purchased At” field.

Events structure

- CustomerCreated
    - Posted whenever a new customer is created
    - Data passed:
        - Email
        - Address

- AddressUpdated
    - Posted whenever a customer’s address is updated
    - Data passed:
        - Customer
- OrderCompleted
    - Posted by external service, subscribed by Customers service
    - Data passed:
        - Order
            - CustomerEmail
            - OrderCreatedAt
    - Action to handle: Update PurchasedAt property of a customer

Functionalities

- Create customers
- Archive customers
- Update customer address

Tasks

- ☐ Design a database schema
- ☐ Write an application in .NET (.NET Core or .NET 6.0)
- ☐ Add a message broker
- ☐ Write a few example unit and integration tests
- ☐ A nice little README on how to install and run
- ☐ Place the project privately on your github account and share it with *omniscript-dev*
- ☐ **Extra:** Use event sourcing & DDD (20 points)
- ☐ **Extra:** Docker file to set up the whole application with dependencies (10 points)
- ☐ **Extra:** Deploy on a free cloud service (10 points)
- ☐ **Extra:** Add Swagger API documentation (5 points)