MICROPROFILE

Jakarta EE for Microservices

- Web services
  - JAX-RS
- Dependency injection
  - CDI
- Binding
  - JSON-B
- Persistence
  - JPA

Microprofile API

- OpenTracing: Log aggregation
- Open API: generation of client code
- REST client
- Config
- Fault Tolerance
- Metrics: monitoring data
- JWT: session tokens
- Health

67

67

Microservice Advantages

- Smaller codebase
- Good coding practices
  - Loose coupling and high cohesion
- Greater resilience
- Scalability
- Polyglot applications

68

68

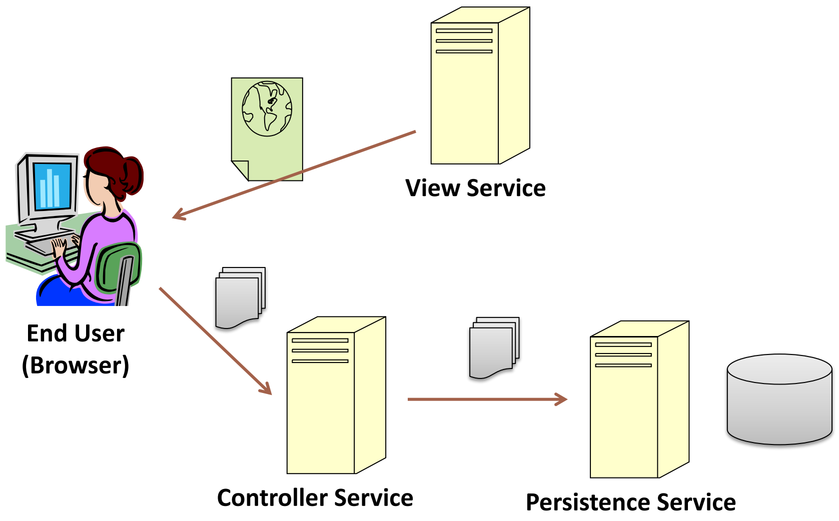
Microservice Disadvantages

- Operational and tooling overhead
- Debugging
- Distributed transactions
  - Rollback and compensating transactions
- Network latency
- Complex interdependencies
- Fallacies of distributed computing
  - Zero latency, infinite bandwidth, etc

69

69

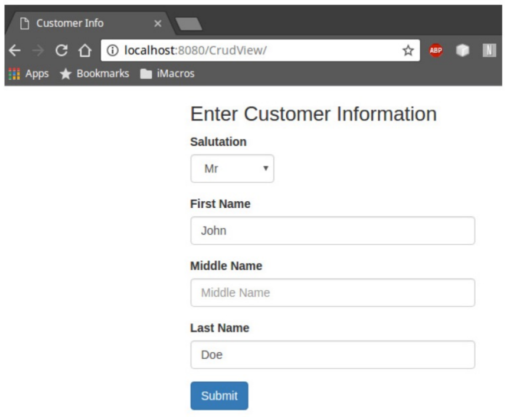
Example



70

70

Example: MVC View



Enter Customer Information

Salutation: Mr

First Name: John

Middle Name: Middle Name

Last Name: Doe

Submit

71

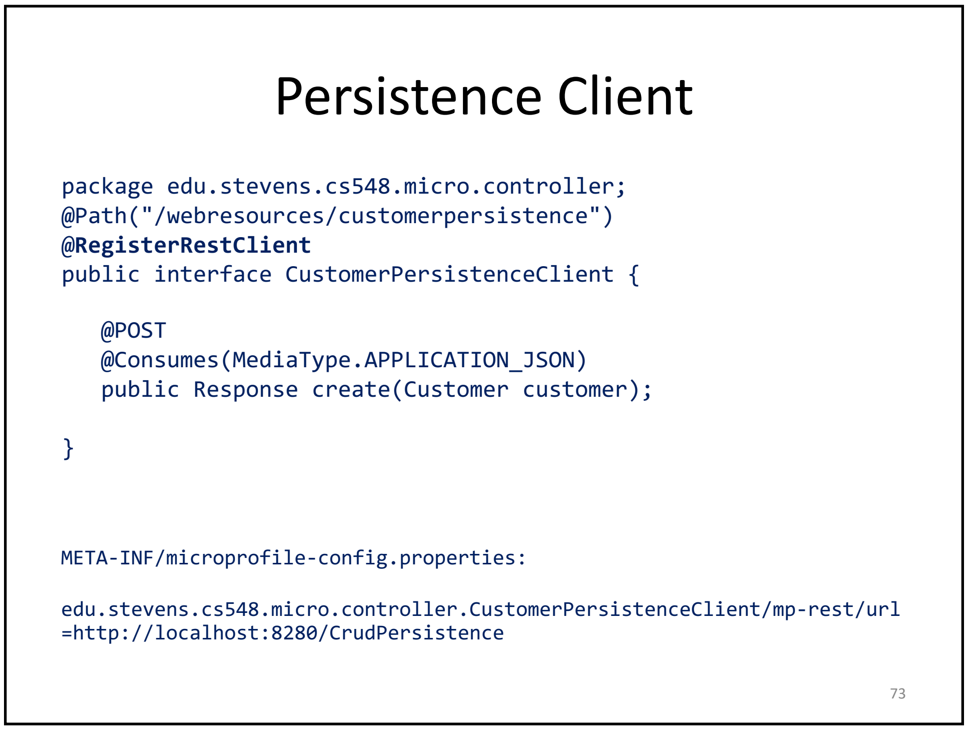
71

Example: MVC Controller

```
@Path("/customercontroller")
public class CustomerControllerService {

   @Inject
   @RestClient
   private CustomerPersistenceClient customerPersistenceClient;

   @OPTIONS
   public Response options() {
      return Response.ok("").header("Access-Control-Allow-Origin",
                                    "http://localhost:8080").build();
   }

   @POST
   @Consumes(MediaType.APPLICATION_JSON)
   public Response addCustomer(Customer customer)  {
      Response persistenceServiceResponse =
         customerPersistenceClient.create(customer);
      ...
   }
}
```

72

72



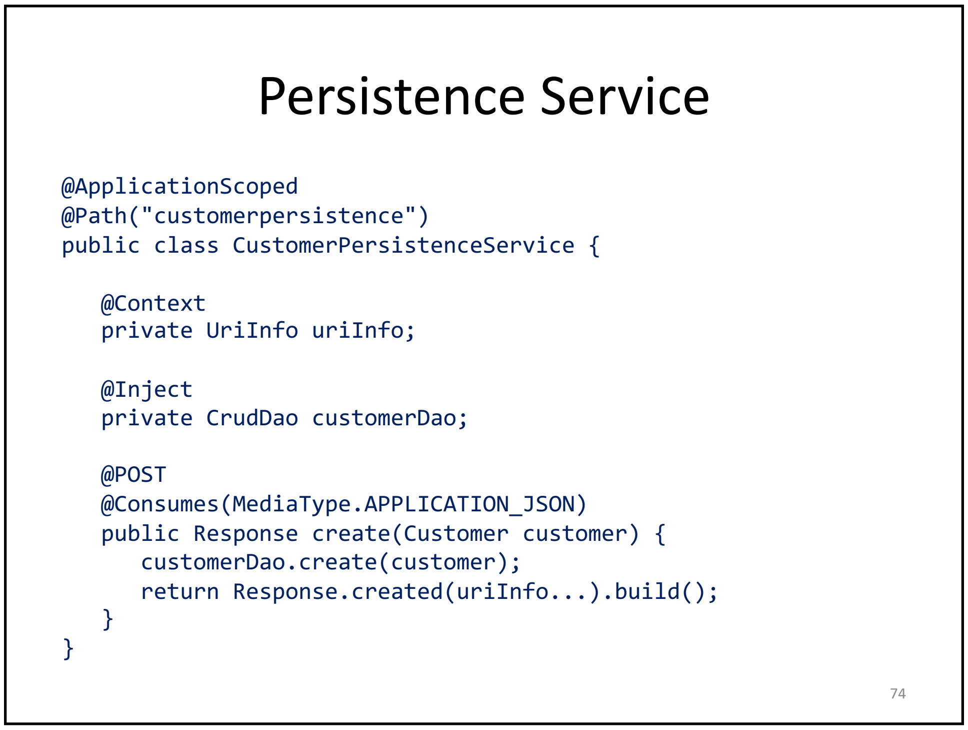
Persistence Client

```
package edu.stevens.cs548.micro.controller;
@Path("/webresources/customerpersistence")
@RegisterRestClient
public interface CustomerPersistenceClient {

   @POST
   @Consumes(MediaType.APPLICATION_JSON)
   public Response create(Customer customer);

}
```

```
META-INF/microprofile-config.properties:

edu.stevens.cs548.micro.controller.CustomerPersistenceClient/mp-rest/url
=http://localhost:8280/CrudPersistence
```

73

73

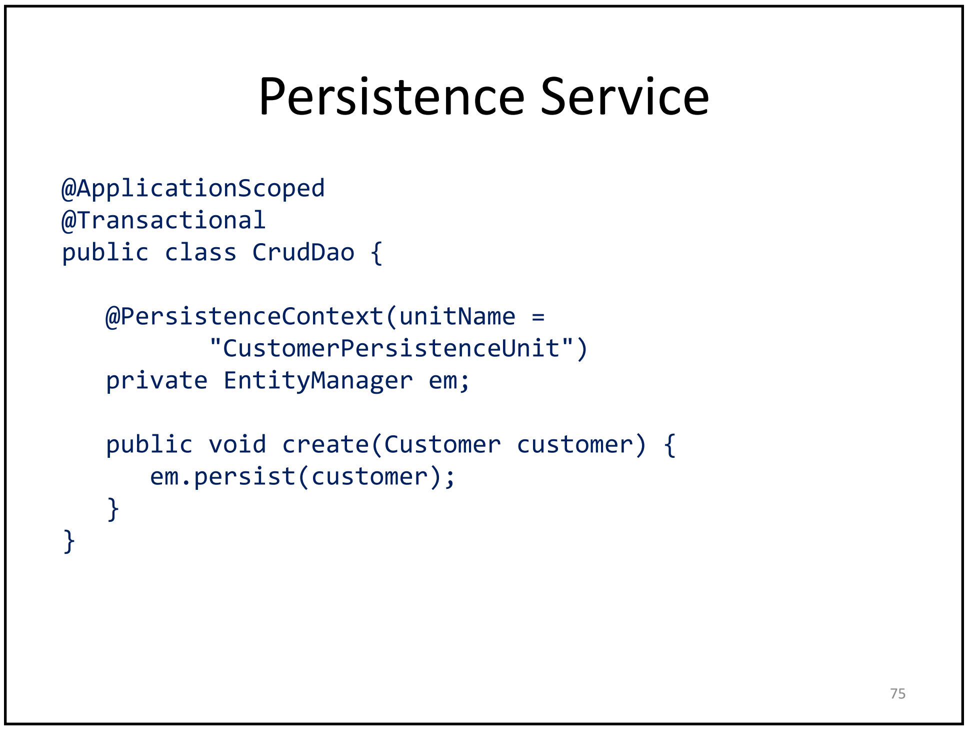


Persistence Service

```
@ApplicationScoped
@Path("customerpersistence")
public class CustomerPersistenceService {

   @Context
   private UriInfo uriInfo;

   @Inject
   private CrudDao customerDao;

   @POST
   @Consumes(MediaType.APPLICATION_JSON)
   public Response create(Customer customer) {
      customerDao.create(customer);
      return Response.created(uriInfo...).build();
   }
}
```

74

74

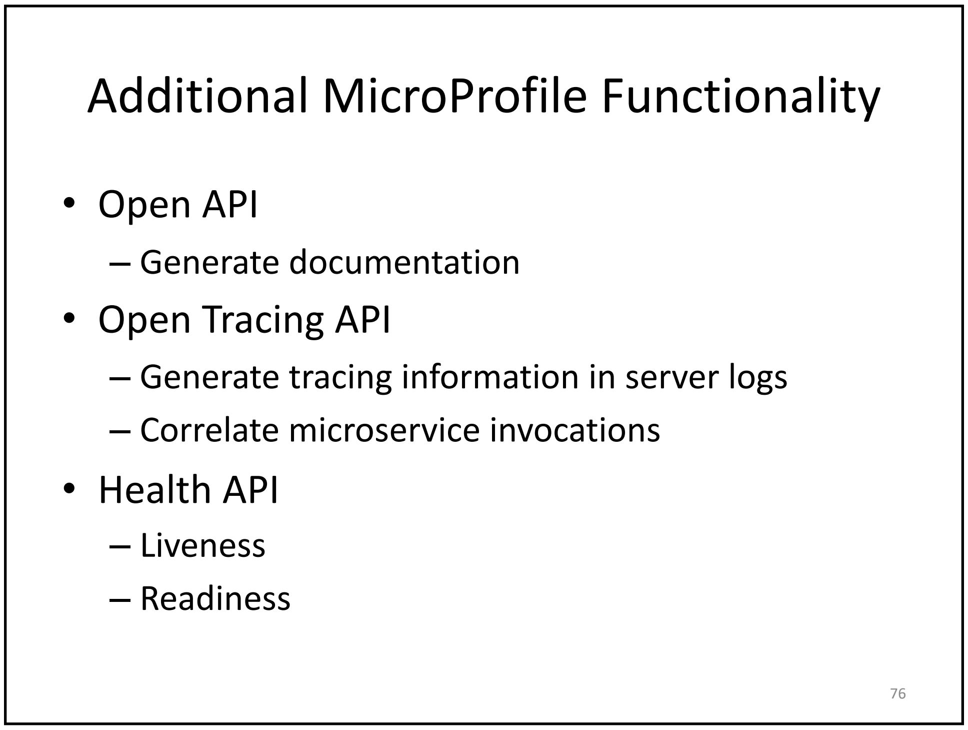


Persistence Service

```
@ApplicationScoped
@Transactional
public class CrudDao {

   @PersistenceContext(unitName =
          "CustomerPersistenceUnit")
   private EntityManager em;

   public void create(Customer customer) {
      em.persist(customer);
   }
}
```

75

75

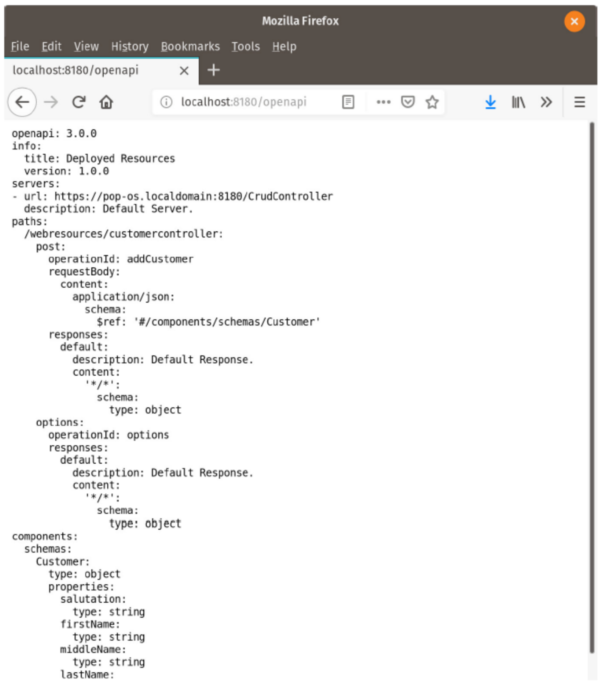


Additional MicroProfile Functionality

- Open API
  - Generate documentation
- Open Tracing API
  - Generate tracing information in server logs
  - Correlate microservice invocations
- Health API
  - Liveness
  - Readiness

76

76



```
openapi: 3.0.0
info:
  title: Deployed Resources
  version: 1.0.0
servers:
- url: https://pop-os.localdomain:8180/CrudController
  description: Default Server.
paths:
  /webresources/customercontroller:
    post:
      operationId: addCustomer
      requestBody:
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Customer'
      responses:
        default:
          description: Default Response.
          content:
            '*/*':
              schema:
                type: object
    options:
      operationId: options
      responses:
        default:
          description: Default Response.
          content:
            '*/*':
              schema:
                type: object
components:
  schemas:
    Customer:
      type: object
      properties:
        salutation:
          type: string
        firstName:
          type: string
        middleName:
          type: string
        lastName:
```

77

77

Trace Spans

```
{ “operationName”: “processWebserviceRequest”,
  “spanContext”: {“spanId”: “f849ed14-d546-4e82-8955-a2871ff8083b”,
                  “traceId”: “5f4c72fb-a791-4a9f-9a5d-8a78fa600a1d” },
  “spanTags”: [
     { “referer”: “[http://localhost:8080/CrudView/]” },
     { “origin”: “[http://localhost:8080]” },
     { “Method”: “POST” },
     { “URL”:
“http://localhost:8180/CrudController/webresources/customercontroller/” },
     { “ResponseStatus”: “200” },
     { “host”: “[localhost:8180]” }
   ],
  “references”: [
     { “spanContext”: {
          “spanId”: “721fb7d3-cbfb-4548-9ea4-eb124d107d24”,
          “traceId”: “5f4c72fb-a791-4a9f-9a5d-8a78fa600a1d” },
       “relationshipType”: “ChildOf” }
  ]
},
```

78

78

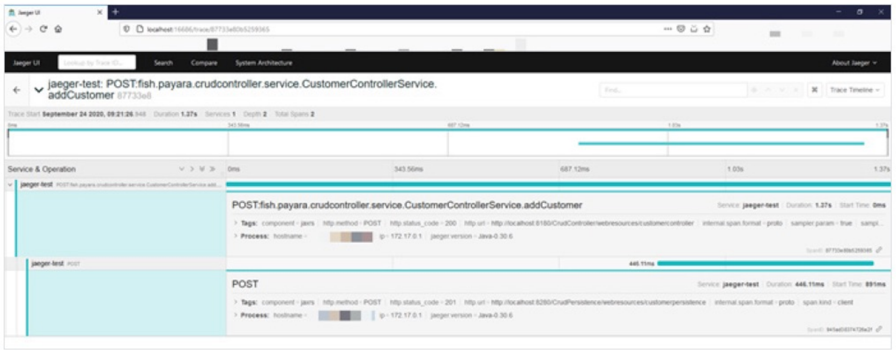
Trace Spans

```
{ “operationName”:
“POST:edu.stevens.cs548.micro.crudcontroller.service.CustomerControllerServi
ce.addCustomer”,
  “spanContext”: { “spanId”: “1694-5b76-794d-44e3-8052-e91e2e572f6f”,
                  “traceId”: “5f4c72fb-a791-4a9f-9a5d-8a78fa600a1d” },
  “spanTags”: [
     { “component”: “jaxrs” },
     { “span.kind”: “server” },
     { “http.url”:

“http://localhost:8180/CrudController/webresources/customercontroller/” },
     { “http.method”: “POST” }
  ],
  “references”: [
     { “spanContext”: {
          “spanId”: “f849ed14-d546-4e82-8955-a2871ff8083b”,
          “traceId”: “5f4c72fb-a791-4a9f-9a5d-8a78fa600a1d”
        },
       “relationshipType”: “ChildOf” }
  ]
]
```

79

79

Jaeger Tracing



jaeger-test: POST:fish.payara.crudcontroller.service.CustomerControllerService.addCustomer 87733e8

Trace Start September 24 2020, 09:21:26.948 Duration 1.27s Services 1 Depth 2 Total Spans 2

0ms 343.56ms 687.12ms 1.03s 1.37s

jaeger-test POST:fish.payara.crudcontroller.service.CustomerControllerService.add...

POST:fish.payara.crudcontroller.service.CustomerControllerService.addCustomer   Service: jaeger-test | Duration: 1.27s | Start Time: 0ms

Tags: component - jaxrs | http.method - POST | http.status_code - 200 | http.url - http://localhost:8180/CrudController/webresources/customercontroller | internal.span.format - proto | sampler.param - true | sampl...

Process: hostname - [redacted] | ip - 172.17.0.1 | jaeger.version - Java-0.30.6

jaeger-test POST   446.11ms

POST   Service: jaeger-test | Duration: 446.11ms | Start Time: 891ms

Tags: component - jaxrs | http.method - POST | http.status_code - 201 | http.url - http://localhost:8280/CrudPersistence/webresources/customerpersistence | internal.span.format - proto | span.kind - client

Process: hostname - [redacted] | ip - 172.17.0.1 | jaeger.version - Java-0.30.6

80

80

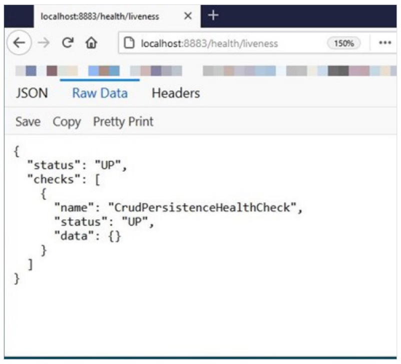
Persistence Service

```
@ApplicationScoped
@Liveness
public class CrudPersistenceHealthCheck implements HealthCheck {

   @Inject
   private CrudDao crudDao;

   @Override
   public HealthCheckResponse call(){
      boolean valid = crudDao.checkDatabaseConnection();
      if (valid) {
         return HealthCheckResponse.named(
            CrudPersistenceHealthCheck.class.getSimpleName()).up().build();
      } else {
            CrudPersistenceHealthCheck.class.getSimpleName()).down().build();
      }
   }
}
```

81

81

Health


localhost:8883/health/liveness
localhost:8883/health/liveness
150%
JSON
Raw Data
Headers
Save
Copy
Pretty Print

```
{
  "status": "UP",
  "checks": [
    {
      "name": "CrudPersistenceHealthCheck",
      "status": "UP",
      "data": {}
    }
  ]
}
```

82

82