ANZ Wholesale Engineering – Sample Project

Spring boot based REST APIs are created based on provided requirements. The application is deployed into AWS and made available via HTTPs endpoints.

Features are present to obtain paginated and filtered information. H2 in-memory database is used to store demo data that supports the APIs. Method calls are also cached so that multiple calls to database are not made. The APIs are documented using Swagger UI. JPA interface based projections are used to respond with only required attributes.

Actuator endpoints are exposed to determine application health, information and metrics.

- API to fetch accounts that belong to a specific user - **/users/{id}/accounts**
- API to fetch transactions for the provided account - **/accounts/{accountNumber}/transactions**

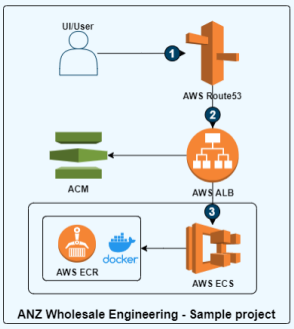
How to access the application and underlying code

The code of application is checked-in to Github as a public repository, the repository can be accessed by using the following link,
https://github.com/Uttam92477/wholesale

The application endpoints are directly accessible using the following links
- https://anzwholesale.learnerdev.com/users/1/accounts
- https://anzwholesale.learnerdev.com/accounts/a1/transactions

The application can also be started locally. The commands and prerequisite environment details are provided in the application documentation in README.md file

Solution Design



Flow	Description
1	The API is exposed through Route53 using a custom sub-domain. UI/API user can access the API by making a HTTPs call to the endpoint
2	The Route53 A record routes traffic to an application load balancer. The application load balancer uses certificate provided by ACM to enable HTTPs listener
3	The load balancer sends the traffic to the API hosted as AWS ECS service by means of a target group

Key Features	Description	Technology/ Framework used
HTTPs endpoints	The application is deployed in AWS and HTTPs endpoints are available to be consumed. ACM Certificate is attached to load balancer lister to enable HTTPs.	ACM, AWS ECS, AWS ECR, AWS ALB, Route53
Pagination	Request parameter based page querying is available. Supported queries page, size	Spring boot - Pageable
Filters	Transaction type filter is available for querying transactions. Query param - transactionType	Spring boot
Actuator endpoints	The default health, info and metrics actuator endpoints are enabled and more can be enabled if required	Spring boot actuator
Swagger	Application specification is available both as swagger JSON spec and swagger UI at /swagger-ui and /v2/api-docs	springfox swagger
H2 DB and seeding	Automatic demo database is created at startup and required data is seeded so that demo works as expected	H2 DB, Spring JPA
Cache	The method responses are cached so that we don't query DB frequently	Spring boot – Cacheable

ANZ Wholesale Engineering - Sample Project

The application provides the backend layer to support UI functionality to display accounts that belong to a user and to display transactions made for a specific account. The application is implemented using latest spring boot frameworks and components. The application is deployed into AWS cloud infrastructure and the endpoints are available to be consumed directly using the links mentioned further in this document. This spring boot application can also be started locally.

How to run the application locally

1. Ensure Java is present in the machine. For development OpenJDK 11 is used
2. Download/clone this repository
3. Create the application JAR file using `./mvnw package`
4. Run the application `java -jar target/wholesale-engineering-demo-0.0.1-SNAPSHOT.jar`
5. All the endpoints mentioned above will now be available with address `http://localhost:8080`

Accessing the application endpoints

Description	Direct endpoint	Localhost endpoint
User accounts	https://anzwholesale.learnerdev.com/users/1/accounts	http://localhost:8080/users/1/accounts
Account transactions	https://anzwholesale.learnerdev.com/accounts/a1/transactions	http://localhost:8080/accounts/a1/transactions
Swagger UI	https://anzwholesale.learnerdev.com/swagger-ui/#/	http://localhost:8080/swagger-ui/#/
Swagger JSON spec	https://anzwholesale.learnerdev.com/v2/api-docs	http://localhost:8080/v2/api-docs
Accounts with page options	https://anzwholesale.learnerdev.com/users/1/accounts?page=1&size=2	http://localhost:8080/users/1/accounts?page=1&size=2

Description	Direct endpoint	Localhost endpoint
Transactions with page options	https://anzwholesale.learnerdev.com/accounts/a1/transactions?page=1&size=5	http://localhost:8080/accounts/a1/transactions?page=1&size=5
Transaction type filter	https://anzwholesale.learnerdev.com/accounts/a1/transactions?transactionType=Credit	http://localhost:8080/accounts/a1/transactions?transactionType=Credit

AWS Deployment steps using aws-cli, docker and mvn

The steps are documented here just to showcase my skills in AWS and need not be performed to access direct URLs mentioned above.

1. Get AWS ECR credentials `aws ecr get-login-password --region ap-southeast-2 | docker login --username AWS --password-stdin ${AWSAccountNumber}.dkr.ecr.ap-southeast-2.amazonaws.com`
2. Package the application `./mvnw package`
3. Build `docker build -t wholesale-engineering-demo .`
4. Tag `docker tag wholesale-engineering-demo:latest ${AWSAccountNumber}.dkr.ecr.ap-southeast-2.amazonaws.com/wholesale-engineering-demo:latest`
5. Push image to ECR `docker push ${AWSAccountNumber}.dkr.ecr.ap-southeast-2.amazonaws.com/wholesale-engineering-demo:latest`
6. Update service to spin up 1 task `aws ecs update-service --cluster anz-wholesale-demo --service backend --force-new-deployment --desired-count 1`
7. Update service to ramp down to 0 task `aws ecs update-service --cluster anz-wholesale-demo --service backend --force-new-deployment --desired-count 0`