*ENG2 - report*

Y3877930

1.1 Architecture

Define the overall architecture using recognised notations (e.g. UML component/deployment diagrams or C4 diagrams). Justify how the architecture can scale with increasing user demands, and be adapted to new requirements in the future (e.g. a recommendation system). [15 marks, max 2 pages]

The architecture used is a microservice architecture rather than a Monolithic architecture. As each microservice can be deployed independently and therefore scaled independently, particularly if there is increased user demands then microservices handling core functions such as authentication or main data throughput can be scaled both horizontally (adding more docker containers for example) or vertically (increasing the CPU power or memory available). Furthermore, each microservice can be worked on separately by different teams as each is responsible for a different function and is only loosely coupled, so this can increase how quickly additional features can be added to a system.

The database choice of MariaDB and docker compose means that it can be scaled horizontally in the same with as the microservices, adding additional nodes to the cluster would mean that the MariaDB could handly more I/O operations at once. Furthermore, Kafka similarly allows scaling, it currently hads 3 nodes but more can easily be added. Kafka handles clustering very easily, having one leader node and the rest are in eventual consistency.

In terms of adding new features to the service, the microservice architecture means that to add new features, a new microservice can be added to the architecture. The new microservice can consume and produce events and do api calls without effecting existing microservices and features.

A blue silhouette of a person

Description automatically generatedA diagram of a company

Description automatically generated

Level 2: Containers

Key

Level 1: Context

A screenshot of a computer screen

Description automatically generated

A diagram of a company

Description automatically generated

Level 3: Components (Same for all 3 microservices)

1.2 Microservices

1.3 Containerisation

Discuss how the solution can scale up to larger numbers of users, and be resilient to failures (e.g. of a container, or a node)