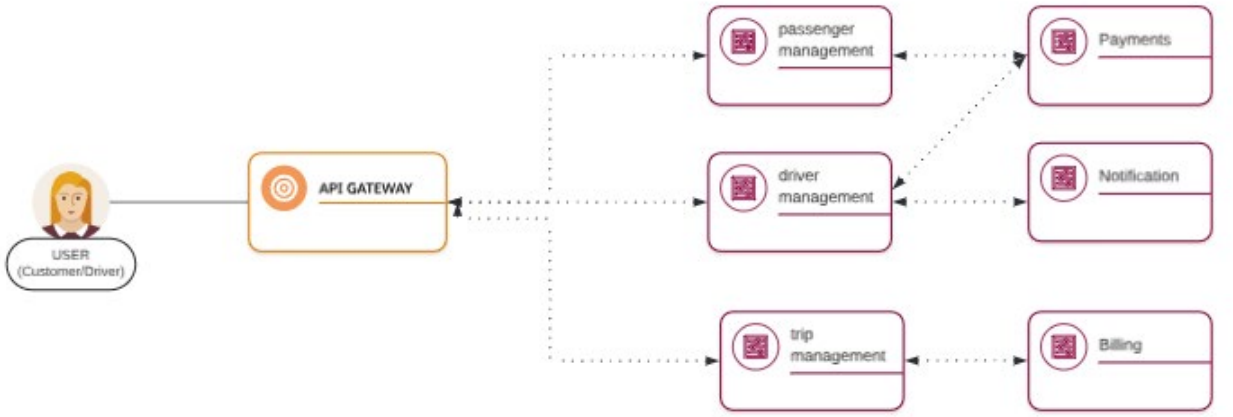
Lab2, Microservices at Uber

Sung Hyun Hong



<Microservices at Uber>

Uber had a monolithic architecture and handled all the features with a single framework before introducing microservices. However, as they expanded their business worldwide, they decided to change their architecture.

Firstly, Uber introduced the API Gateway that users communicate with. The API gateway is connected to different management microservices, such as passenger management, driver management, and trip management. These management microservices, in turn, communicate with other microservices, such as notification and billing.

By breaking apart the single framework into multiple independent deployable modules (microservices), they improved scalability and flexibility and also gained easier maintenance.

One potential drawback of Uber's new microservices architecture is that it could lead to consistency issues among its microservices. For example, if a microservice modifies a schema that is shared with other microservices, it could cause issues for all the microservices that reference that schema. However, this drawback is also applicable to the previous monolithic architecture, as it also requires updates for the entire codebase if there is any change in the schema.

Reference: https://medium.com/edureka/microservice-architecture-5e7f056b90f1