API types

Each backend microservice provides some kind of API to communicate with the external world. It could be HTTP API, gRPC, Kafka event streams, messaging, or some other protocol. Microservices are grouped into technical domains to encapsulate internal details and simplify the overall landscape.

Depending on the purpose and scope of usage API is divided into 3 classes with different requirements and conventions.

Public API

This API is exposed from the API platform for external usage in different channels (mobile applications, web applications, IVR, external systems, etc.).

Challenges and problems

- API could be used by many clients, some even external to the bank, so API contracts must be very reliable;
- mostly this class of API is consumed from outside of the Trust zone so authorization and security measures are mandatory;
- many clients are based on common domain standards, so API must not have internal systems specifics and operates only with the public domain model.

Requirements and conventions

- API strictly follows one of the industry domain and RBI Group standards (Marqeta, Mambu, etc.);
- API is well documented (methods, params, data structures, error codes, etc.) and live documentation is exposed in the form of OpenAPI specification;
- all endpoints are protected with authorization at the user or service level (at the service level accessible scope is defined in the Auth Server);
- any changes in the API must be either 100% backward compatible or versioned (recommended number of concurrent versions is 2);
- API is exposed from the domain via a dedicated API gateway (or ingress controller on the dedicated domain name);
- to use API from a system exposed to the Internet additional component (some kind of API Gateway) is needed in the DMZ zone to implement security measures and prepare API for the particular channel.

Backend API

This API is designed to be used by other backend services from the same or other domains.

Challenges and problems

- API could be used by many clients, so API contracts are hard to control and manage.

Requirements and conventions

- API is well documented (methods, params, data structures, error codes, etc.) and live documentation is exposed in the form of OpenAPI specification;
- all endpoints are protected with authorization at the user or service level (at the service level accessible scope is defined in the Auth Server);
- specifics of internal bank systems could be used in the API;
- any changes in the API must be either 100% backward compatible or agreed with all API clients preliminary (consumer-based contracts could work well);
- API is exposed from the domain via a dedicated API gateway other than one for public API (or ingress controller on the dedicated domain name).

Private API

This API is not exposed outside of the domain and designed only for domain-wide usage.

Challenges and problems

- This API has no specific requirements, so sometimes could be created quickly without careful design and documentation.

Requirements and conventions

- API is well documented (methods, params, data structures, error codes, etc.) and live documentation is exposed in the form of OpenAPI specification;
- specifics of internal bank systems could be used in the API;
- the single version of the API could be supported;

- API usage is restricted to the same domain only.