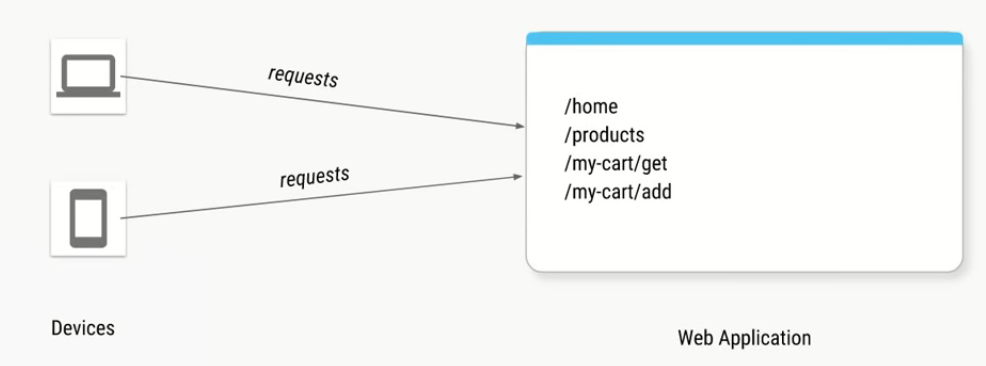
**What is an API Gateway?**

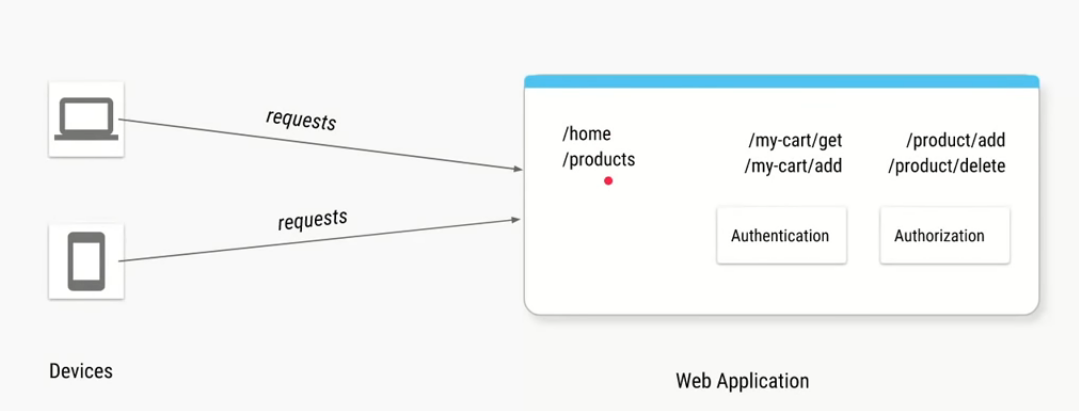
E-Commerce web application

Let us say we are creating an ecommerce application for that we have created a web application which has the above urls.

If you fire a /home url it will give you static HTML page and js files so that you can see the home page.

You have a /products it will give list of all the products in the form of a JSON.

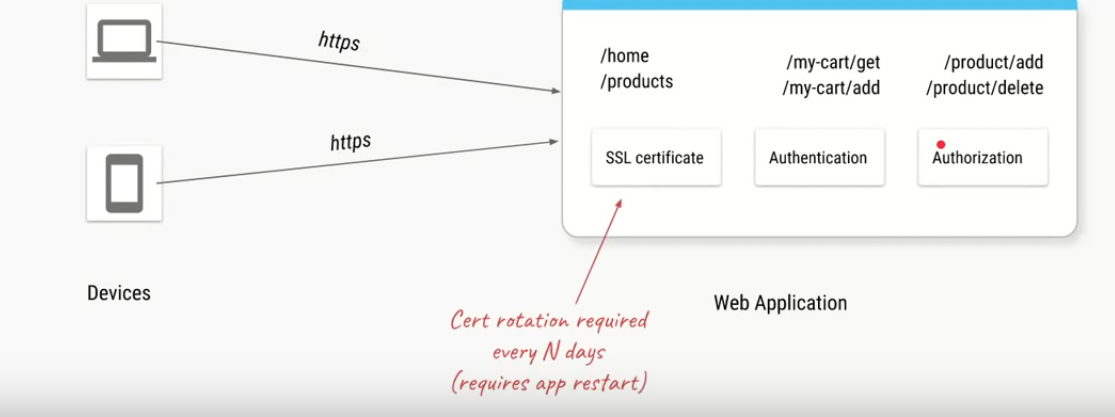
If the user is logged in the user can make the request for getting the cart and adding items to the cart and on the left we have clients which can come from multiple devices types either a desktop or mobile and they will make the requests for these particular URLs



If the User fires req /home or /products even though user is not logged in we want send data back to the user

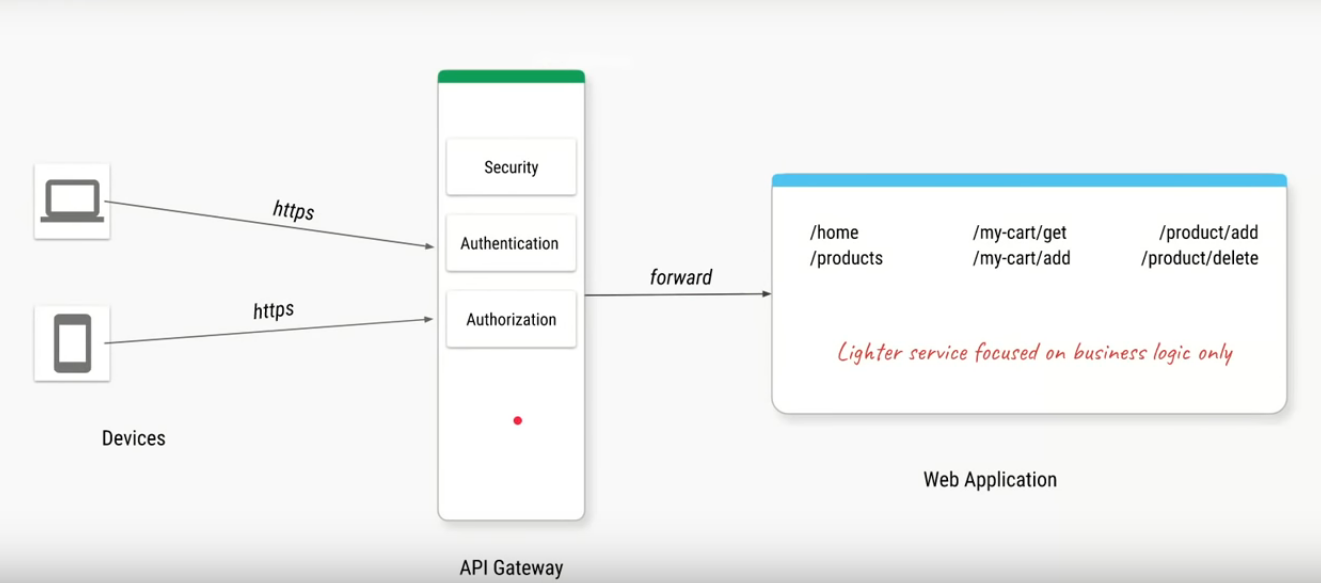
If the user fires any of the operations of the shopping cart which is used to get items from the shopping kart or which is used add items to the shopping kart then we have to ensure that the user is logged in.

We want to ensure that user is authenticated. And similarly we might have some URLs like adding a product or deleting a product which might require a different role, an administrative role and for that we want to authorize whether the user who has signed in and he is trying to access this resource url Is actually an administrator so we need to add the code for authorization also.



Next we want to ensure that the web application is secure so instead of HTTP we want to have HTTPS in our application for that we need to add the SSL certificates and best practice for SSL certificates is to rotate the certificate that means to change the certificate every thirty days or every sixty days.

So basically, your web application these three added components in addition to all the business logic and we want to try to separate out these things which are not business logic related into a separate component and that component is called an API Gate way.



API Gateway:

An API Gateway is a component which acts as an entry point for your APIs so in our case this API gateway are hosted on a web application so it becomes and entry point for our web application

Every time the clients makes request so they will first goto the API Gate way the api gate way will ensure the requests are HTTPS using the SSL certificate they will ensure that the user is authenticated and the API gate way will ensure that the user has the right role if the url requested for a product change and when only these conditions are satisfied that request will be forworded to your web application and that way API gate way will act as guard for any of the request which is coming to your application so it will protect your API from BAD actors and that brings us the first feature.

**Features#1**

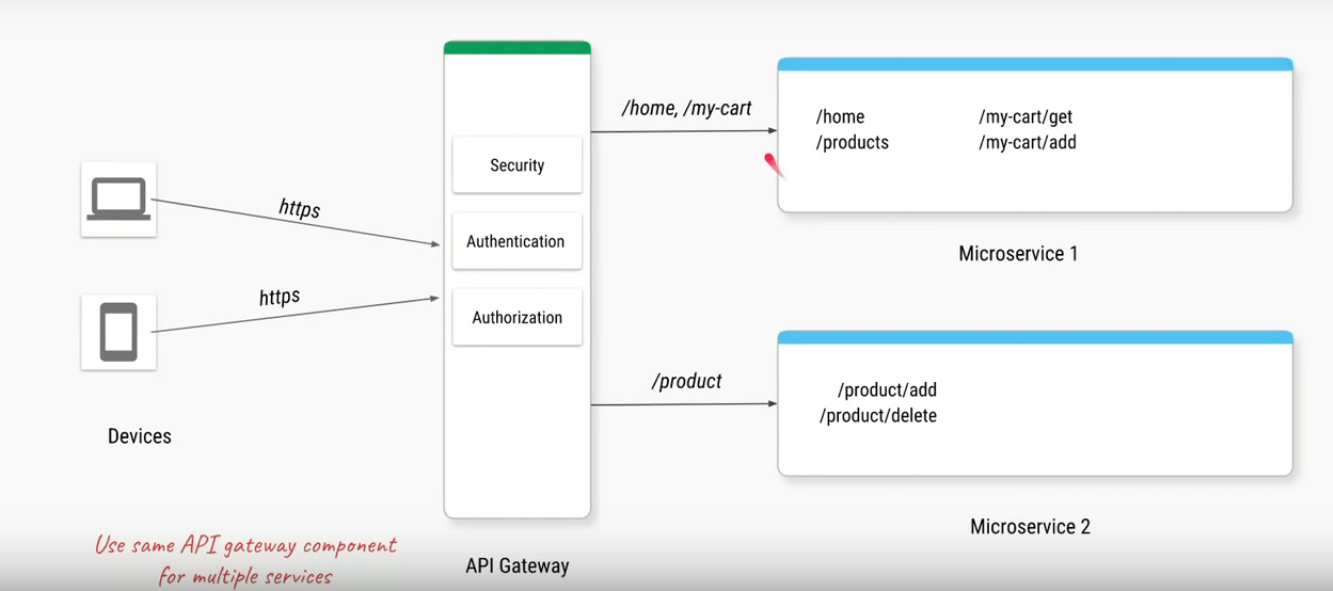
Separate out cross cutting concerns

- Authentication

- Authorization

- SSL termination

- DDoS protection / Throttling



Let us say we convert our single monolith in multiple micro services so if there are any URLs hit from the client for /home /my-cart then it should goto micro service1 but anything related to product then URL says /product then that should goto micro service2 now since all your requests will always come on this API gateway which is a single component your API gate way can decide based on these paths where to route your URL’s either to micro service1 or micro service2 that brings us the feature2.

Feature#2

Separate and consolidate cross cutting concerns across microservice into a single component

-Authentication

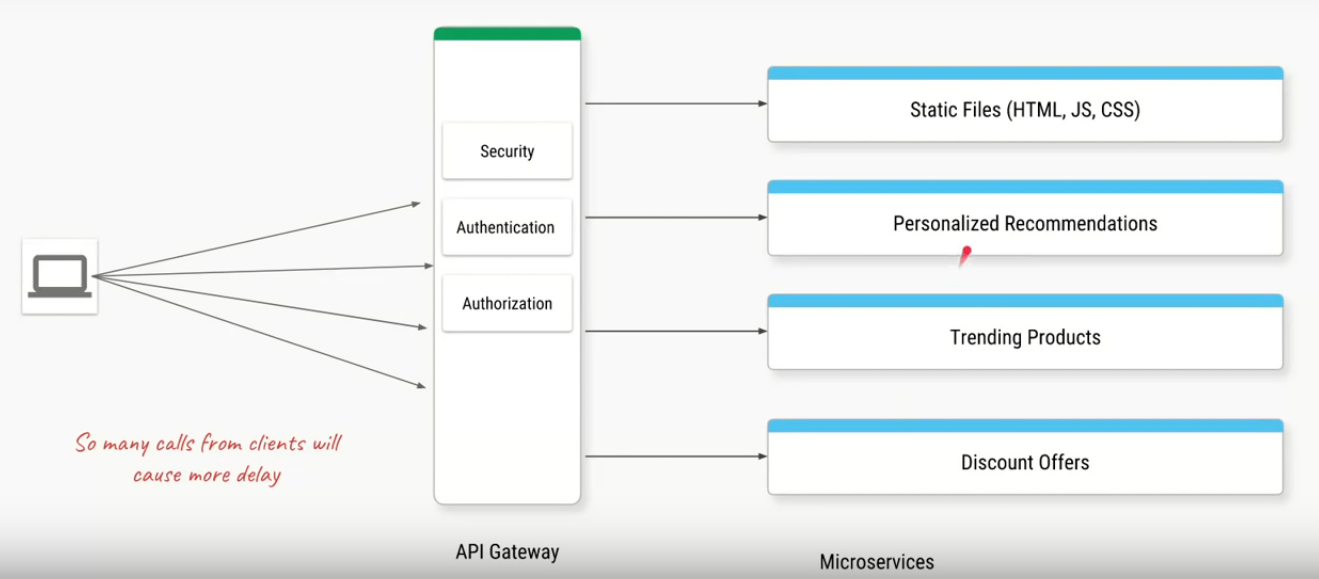
-Authorization

-SSL termination

-DDoS protection / Throttling

-Routing

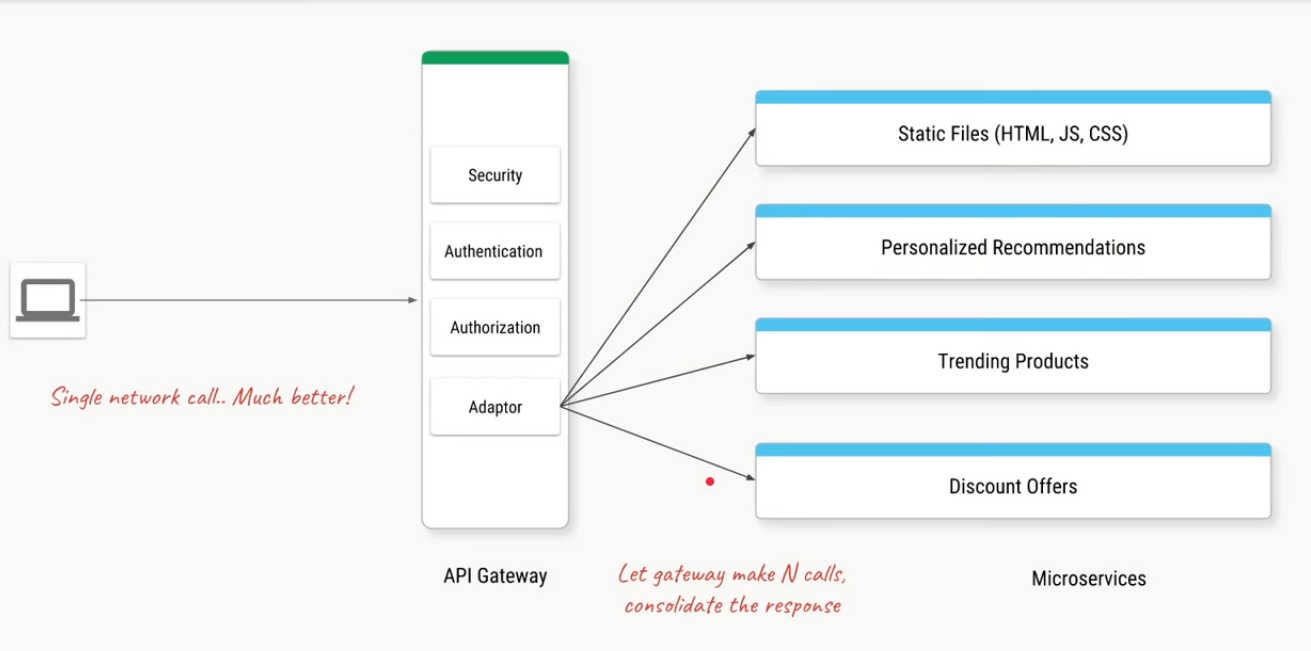
Let us say our e-commers application is more popular we added more services to it.



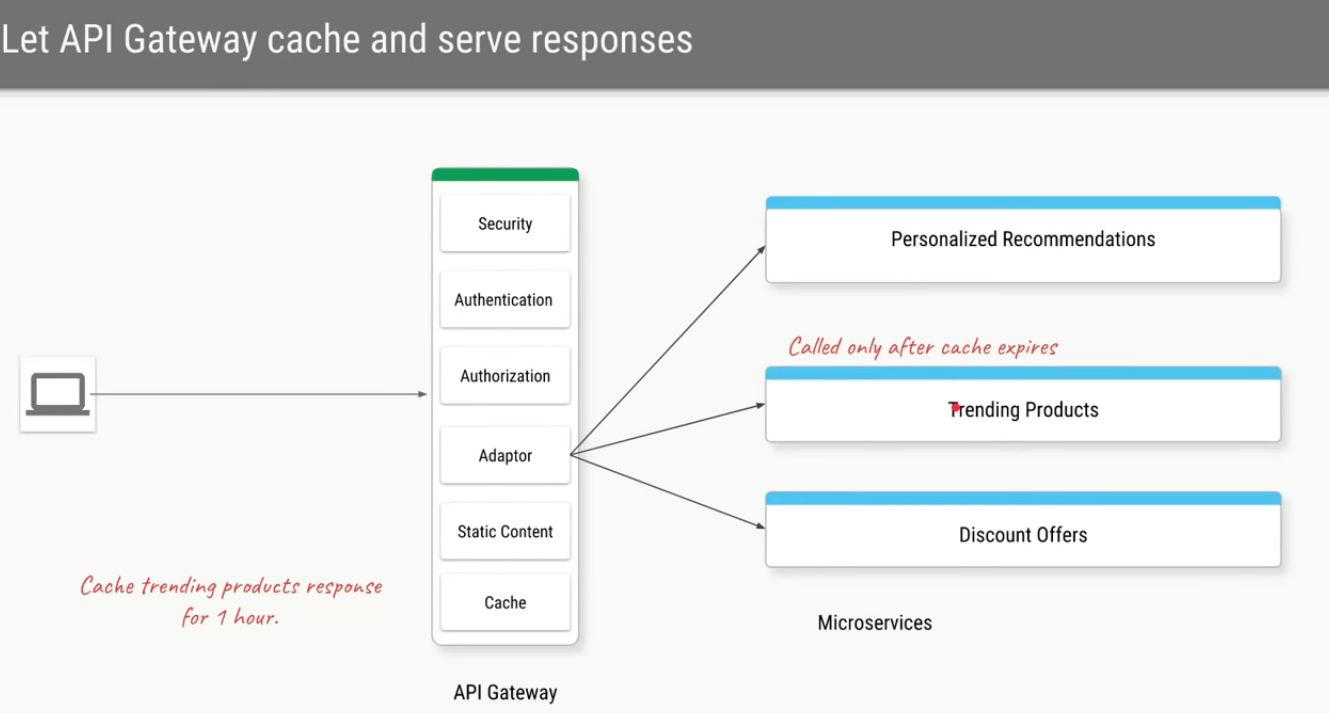
Let us say our e com application becomes more popular and we want to add more features it.

We want to add features where we give personalized recommendations for the logged in user

we give list of all trending products and we also have some discount offers we want to provide the client and let us say all of these features provided by distinct micro services so the client will now first make a call for the home page and will get all the static files and then will make ajax calls which are http calls for the personalization recommendations for trending products and discount offers so now we have total four http calls made by the client to ensure the home page is rendered correctly and is possible that when the application is even grows further we have even more services and the client will have even more number of HTTP calls to even render a single home page so instead the better option is the client will send you only a single call saying I need the home page and we will have some component in API gate way lets us call an adapter which will make all those four calls for the client on behalf of the client it will consolidate the responses from these services and send back single response to the client



That means the response time to render the home page is must faster also since there is one service serving only static files which might not be the best use case for micro service and there is no business logic it is just serving same files again and again we can have a component within the API gateway where if there are any requests for those static files API gateway will it self-return the response and that way we can just decommission this micro service completely.



Let us say we have a service to fetch the trending products

Let us say in our case the trends of those products are recalculated only after one hour so if there are 50 calls with in an hour by the client it will always be forwarded by the API gateway to this micro service and micro service is keep returning the same response again and again.

An API gateway has a feature called response cache where you will give it URL and a threshold time for which it needs to cache the responses.

So for the first time the client makes a call it will forward it to trending products micro service it will get the response it will cache it for an hour

Next time the client sends the same request instead of going back to the service it will return the response from the cache itself. That brings in the feature number 3 for the api gate way.

1. Instead of client making too many network calls we can have the API gate way make all those network calls consolidate the response and send a single response back.

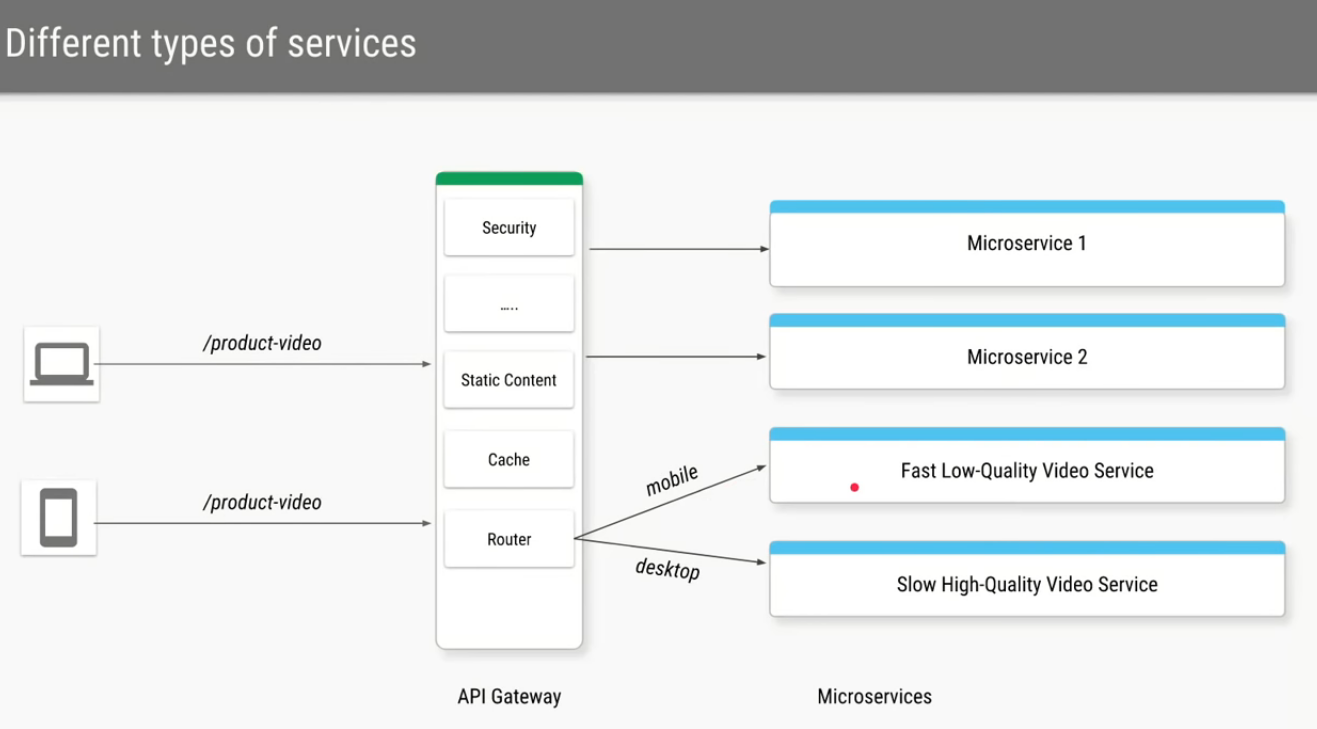
2.To serve the static content and to serve the cached content

Feature#3

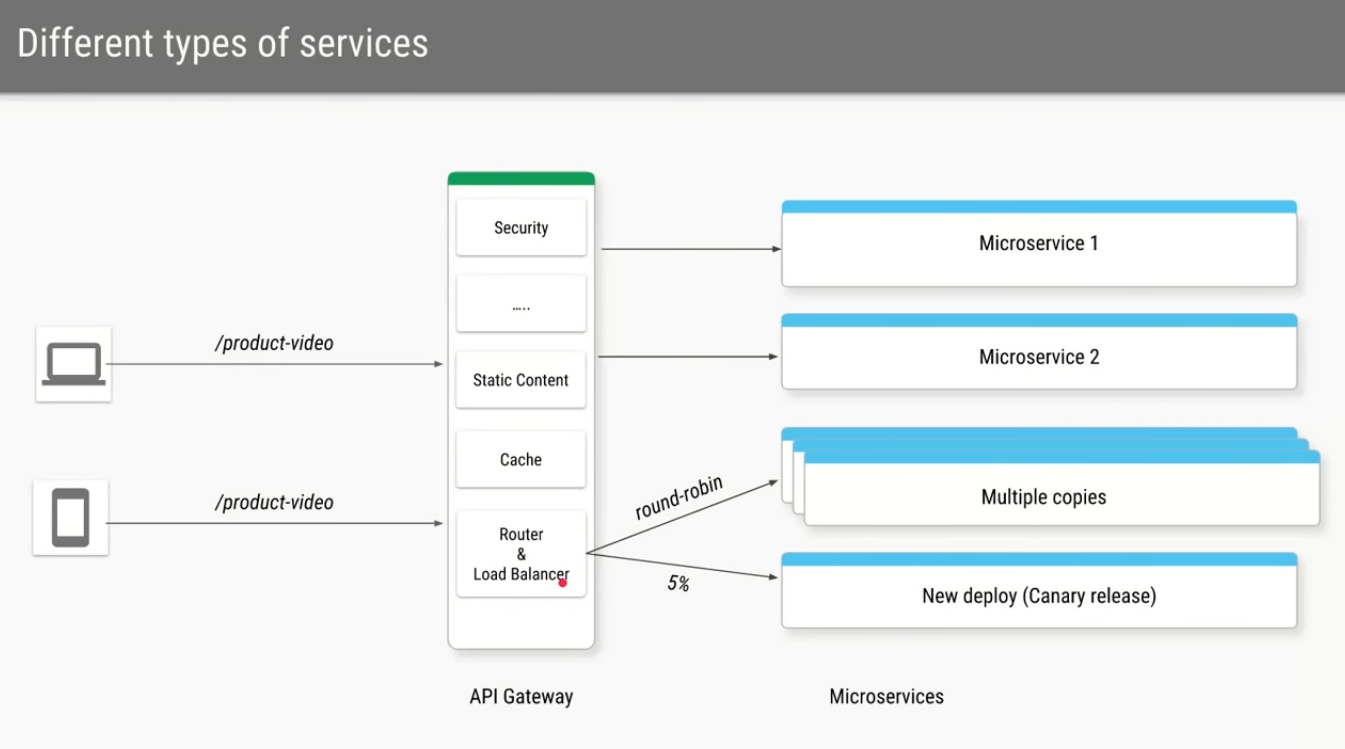
Replacing multiple client calls with single API call.

Also some features of reverse proxy:

* Serving static content
* Caching responses



Let us say we have expose a url called /product-video we will have two services in the back end one service which is responsible for faster low quality video and which will be slow high quality video API gateway we can configure such that request is coming from the mobile and route those request to the fast service and if the request is coming from desktop and we can route that request to hight quality service.



Let us say one of our service become more popular too many API hits to that service and we make multiple copies of that service so in this case the API gateway can also take the responsibility of being a load balancer where the first API call to this service is made to the first copy 2nd call Is made to the second copy and so on and so forth in a round robin fashion and if we have a new version of the same service deployed in API gate way we can configure it so that it sends 5% of the traffic to the new service while reset of the 95% will goto the same old version of that service .

So that brings the fourth features

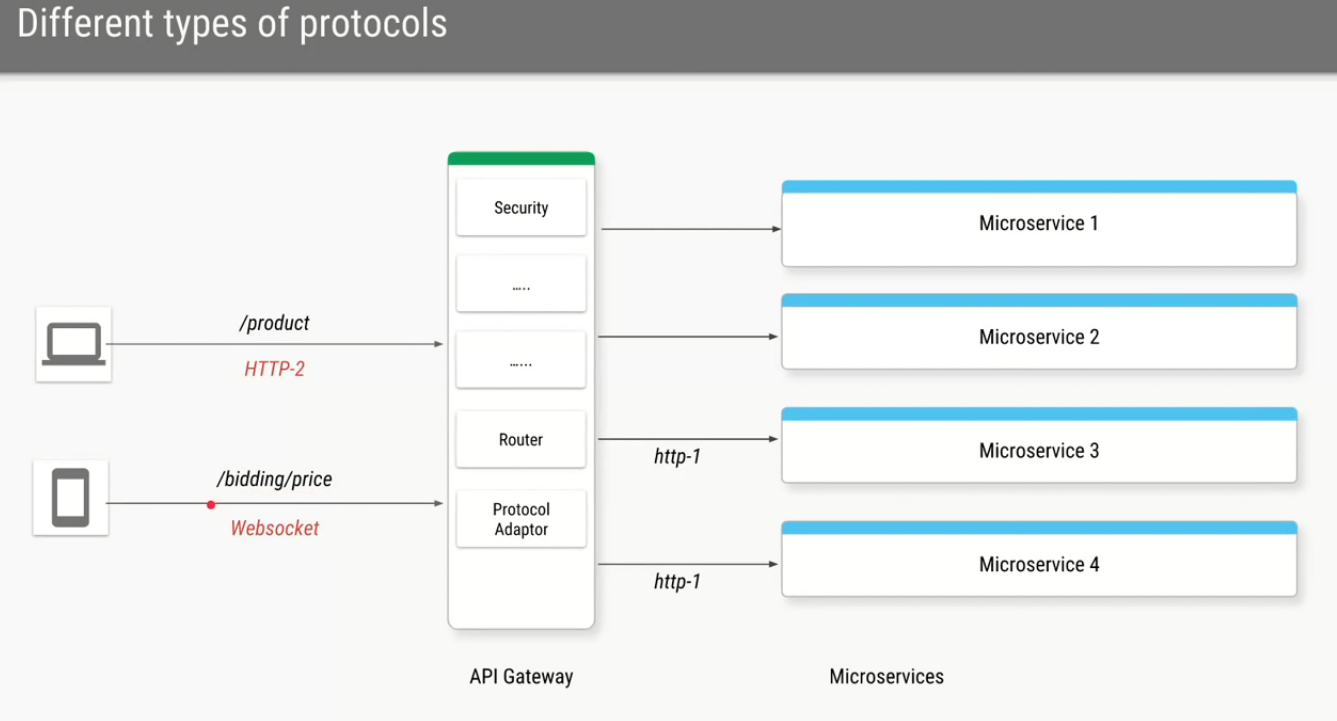
Ie Routing to the right service and routing to the different copies of the service based on whether you are doing load balancing or canary release or even A/B testing

Features#4 Routing based on headers, paths and params etc.

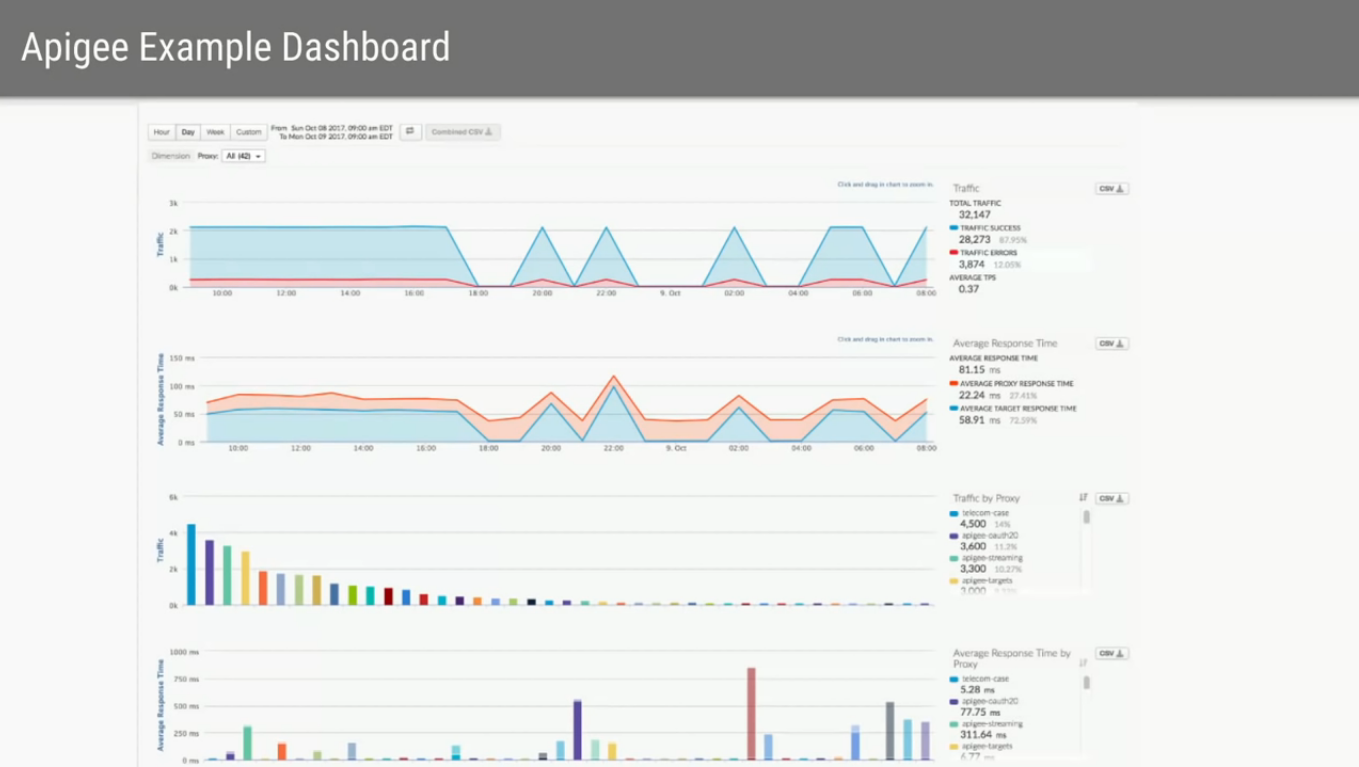
Also, some features of load balancer component

* Load balancing
* A/B testing
* Canary releases

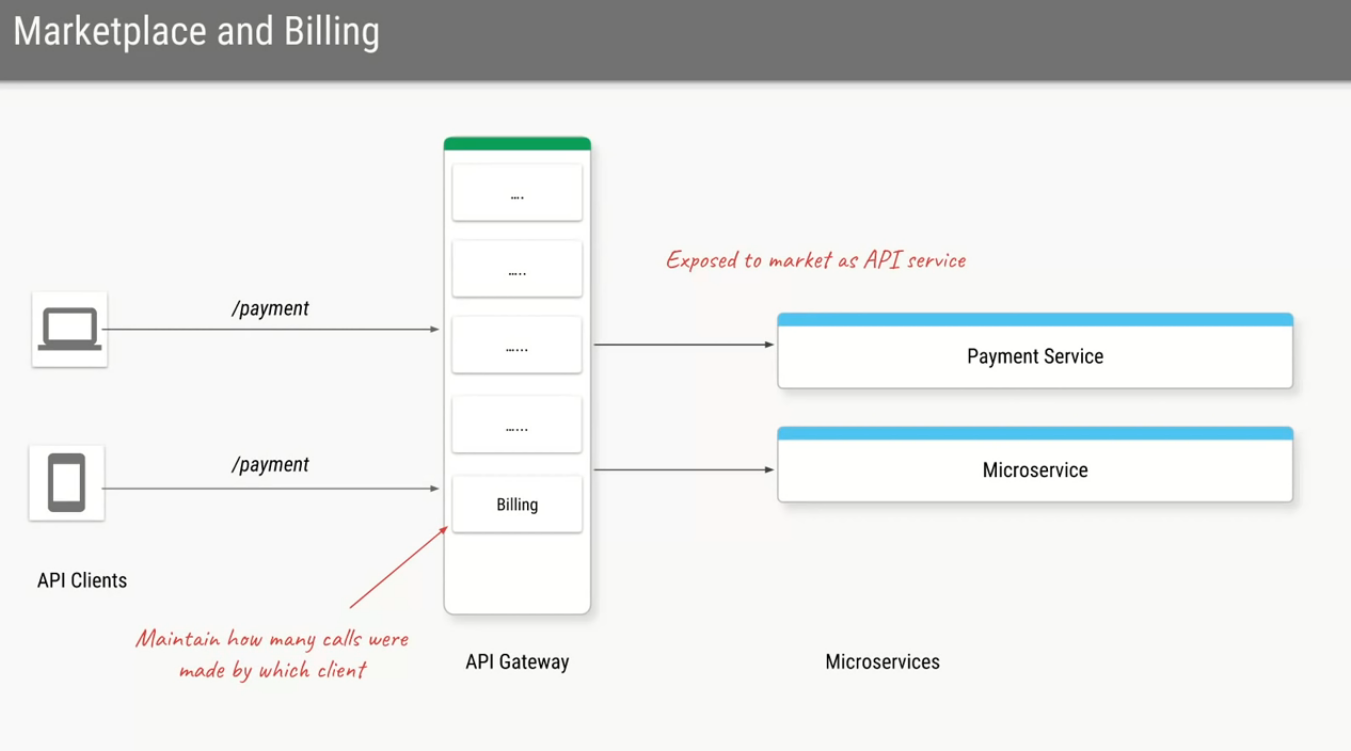
And the next feature of the API gateway deals with different types of protocols.



We want to take the adv of dealing with new protocols such as web-sockets or we want to use newer version of HTTP ie HTTP2 so even if your backend services are not ready are not compatible with HTTP2 or web socket then API gateway can take the responsibility of converting a new protocol to an older protocol



And of course if all these requests are going though a single component API gate way it gives us an extra advantage where we can monitor and log all the requests and the response times where we can calculate the average latencies and most of the API gateways out of the box provieds such dash boards where using the charts we can monitor how your APIs are performing behind the seens



Let us say we developed a micro service which is so reliable and so good that we want to expose it even for other clients so we are actually creating an ecom application but our payment service is very good and we want to expose it as payment gateway such that all the other website can incorporate that so we want to expose the payment service as an api service.

So if we use the API gateway of the cloud providers like AWS can export our payment service as API service to the market place. any client can come in and start using our API Service and based on how many calls they make to your service we will charge them accordingly and the examples of API gateway service so if your are deploying your own application on a virtual machine or on your own data center you can use apache the proxy nginix or spring cloud gateway all these are software’s which you can deploy and configure to run as an API gateway and if you are deploying your applications on the cloud you can use the API gateway service of any of these

Examples:

Self manged:

- apache

- HAproxy

- NGINIX

-Spring Cloud Gateway

Cloud Services

AWS API Gateway

Azure API Gateway

Google Cloud Endpoints

Apigee