



Yapay Zeka I: Veri Bilimi ve Makine Öğrenmesine Giriş Sertifika Programı

Doç. Dr. Taner Arsan

H. Fuat Alsan, PhD(c)

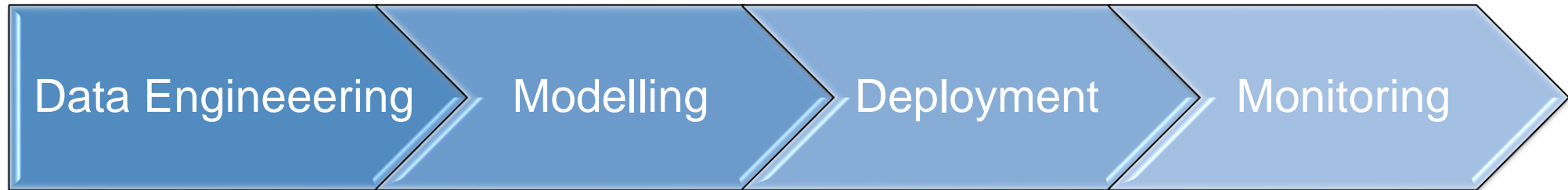
Sena Kılınç, PhD(c)

Konular

- **MLOps Nedir?**
- **Web Servisi Nedir?**
- **API Nedir?**
- **HTTP Rest API**
- **Mikroservis**
- **FastAPI**
- **Docker**
- **MLFlow**
- **Makine Öğrenmesi Modellerinin Web Servisi Olarak Implement Edilmesi**

MLOps

- MLOps (Machine Learning Operations), makine öğrenimi modellerini geliştirme, eğitme, dağıtma, yönetme ve sürdürme sürecini kapsayan bir disiplindir.
 - Daha hızlı ve güvenilir model dağıtımı
 - Artan model performansı
 - Daha iyi model yönetimi
 - Geliştirilmiş işbirliği



Data Engineering

- Veri İşleme ve Hazırlama [Jupyter Notebook]

Modelling

- Model Eğitimi ve Değerlendirmesi [Scikit-Learn, Pytorch, Tensorflow]
- Model Seçimi ve Optimizasyonu

Deployment

- Otomasyon ve CI/CD [Github, Jenkins]
- Model Paketleme ve Dağıtımı [Docker, Kubernetes]

Monitoring

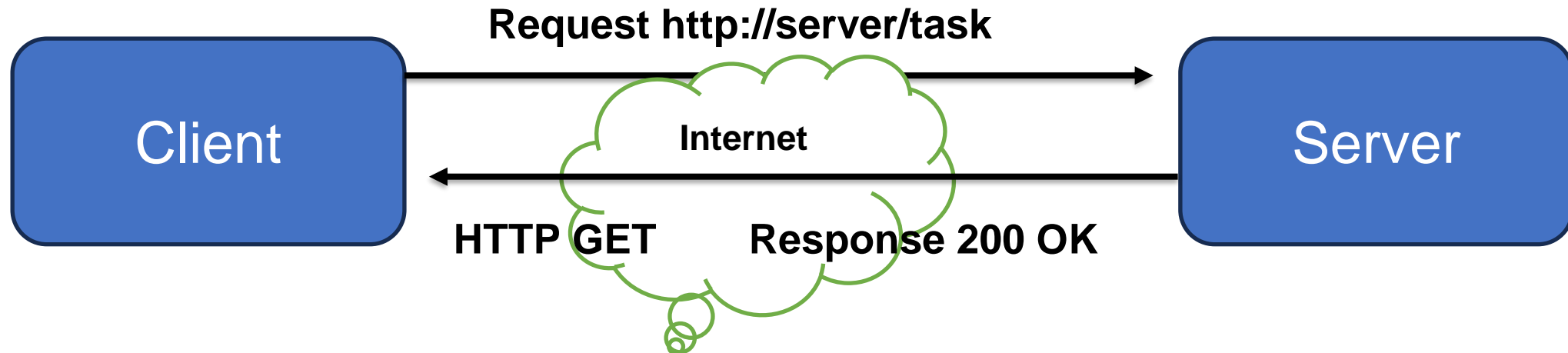
- Model İzleme ve Optimizasyon [MLFlow]
- Model Yönetimi ve Sürüm Kontrolü [Git, Docker]

MLOps

- Starbucks India, müşteri kaybını azaltmak ve satış fırsatlarını artırmak için veri odaklı stratejiler kullanıyor; entegre veri platformu sayesinde hedeflenmiş kampanyalarda **geliri artırıyor** ve **müşteri kaybını azaltıyor**.
- Senko Lojistik Grubu, AI destekli tahminlerle gönderi hacmi doğruluğunu artırmak için MLOps'u benimserken; operasyonel prosedürleri optimize ederek **iş yükünü azaltıyor** ve **tahmin doğruluğunu artırarak** işlemleri düzenliyor.
- PadSquad, reklam **performansını** artırmak ve maliyetleri azaltmak için MLOps'u uygulayarak; otomatik işlemlerle iş yükünü hafifletiyor ve reklam performansını geliştirerek hızlı pazarlama yapma imkanı sağlıyor.
- Philips, Hollanda, sağlık sektöründe: Deneysel izleme ve otomatik belgeleme ile tasarruf edilen **saatler**.
- Ecolab, kimya sektöründe: Model dağıtım **sürelerini** 12 aydan 30-90 güne düşürdü.

Web Servisleri

- Yazılım uygulamalarının farklı platformlar arasında iletişimini sağlayan teknolojilerdir.
- Sunucu tarafından sağlanan ve istemci uygulaması tarafından kullanılan hizmetlerdir.
- Genellikle HTTP protokolü üzerinden iletişim kurarlar ve XML veya JSON gibi veri formatlarını kullanırlar.
- Platform bağımsızdırlar, farklı teknolojilere sahip uygulamalar arasında entegrasyon sağlarlar.



XML ve JSON

```
<bookstore>
  <book category="fiction">
    <title lang="en">Harry Potter</title>
    <author>J.K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="fiction">
    <title lang="en">The Hobbit</title>
    <author>J.R.R. Tolkien</author>
    <year>1937</year>
    <price>19.99</price>
  </book>
</bookstore>
```

```
{
  "bookstore": {
    "book": [
      {
        "category": "fiction",
        "title": "Harry Potter",
        "author": "J.K. Rowling",
        "year": 2005,
        "price": 29.99
      },
      {
        "category": "fiction",
        "title": "The Hobbit",
        "author": "J.R.R. Tolkien",
        "year": 1937,
        "price": 19.99
      }
    ]
  }
}
```

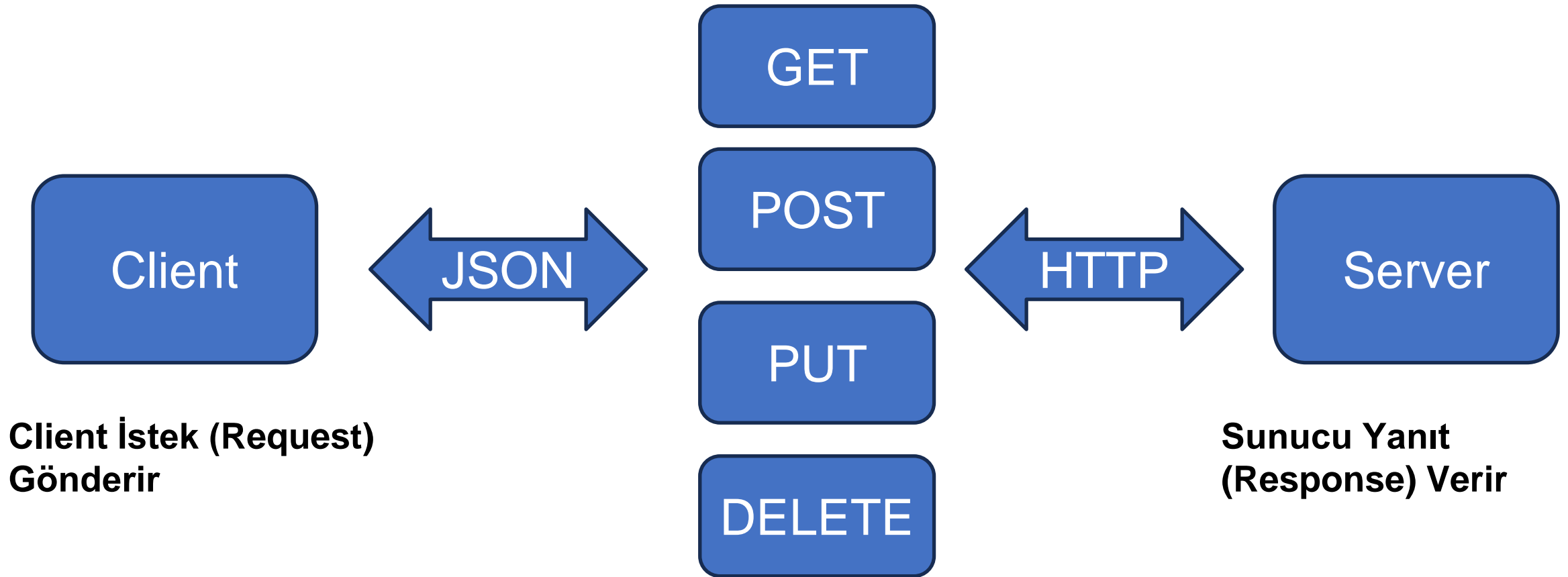
API

- API'ler veya Uygulama Programlama Arayüzleri, yazılım uygulamalarının birbirleriyle iletişim kurmasını sağlayan kurallar ve protokoller bütünüdür.
- API'lar, uygulamaların bilgi alışverişi yapmak ve işlevlerini kullanmak için kullanabileceği yöntemleri ve veri formatlarını tanımlar.
- Bir API, bir uygulamanın başka bir uygulama ile etkileşim kurmasını sağlayan bir protokol kümesidir.

Fark nedir?

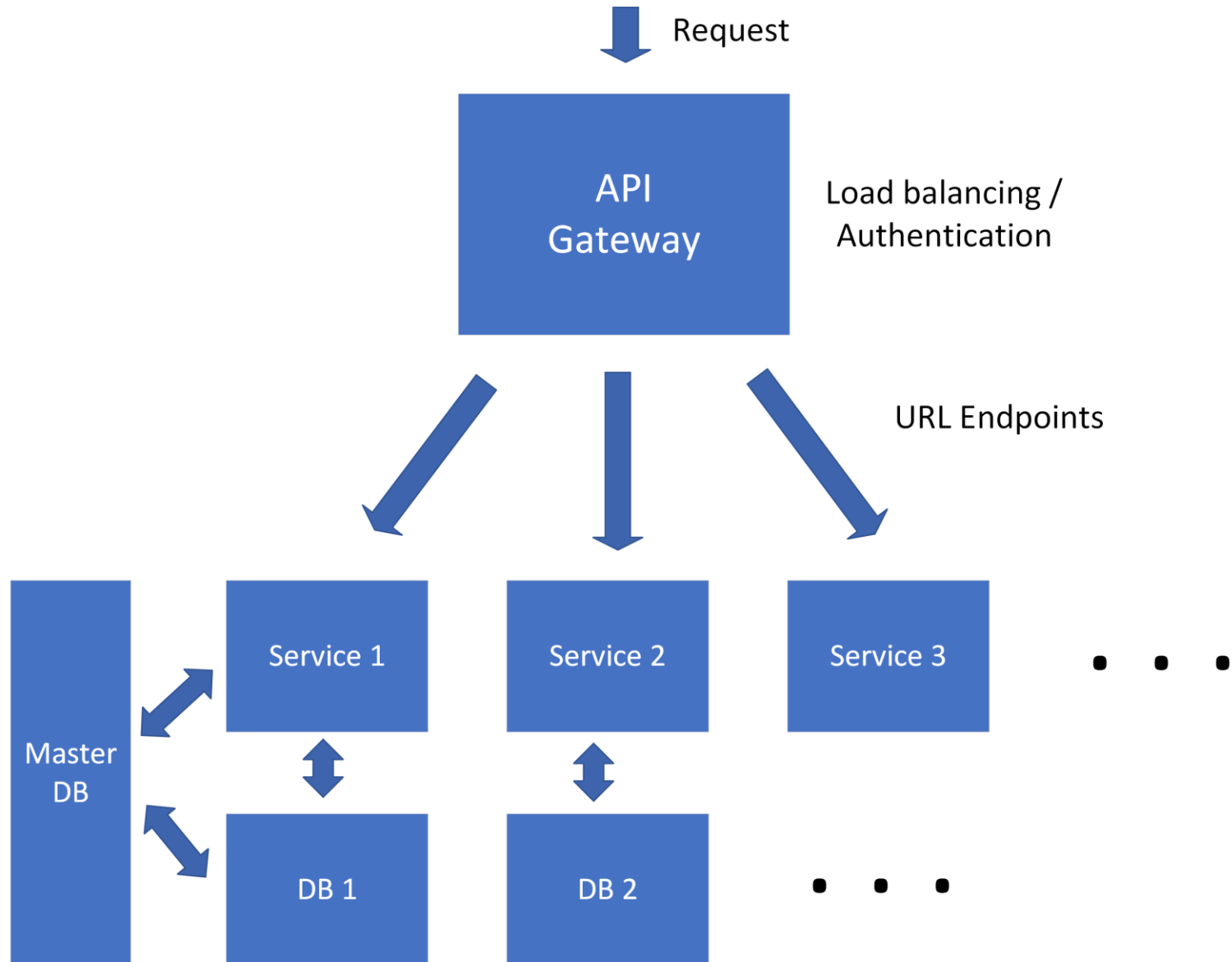
- Web Hizmetleri, bir ağ bağlantısı üzerinden erişilmesi gereken bir API türüdür.
- API'ler, farklı uygulamaların birbirleriyle standart bir şekilde iletişim kurmasına olanak tanıyan uygulama arayüzleridir.
- Tüm Web hizmetleri API'dir ancak tüm API'ler web hizmetleri değildir.
- Web hizmetleri genellikle SOAP ile ilişkilendirilirken API'ler herhangi bir iletişim stilini kullanabilir.
- API'ler verileri JSON veya XML gibi formatlarda döndürebilirken web hizmetleri öncelikle JSON kullanır.
- Web hizmetleri daha ağırdır; API'ler ise akıllı telefonlar gibi sınırlı bant genişliğine sahip cihazlar için uygun, hafif bir mimariye sahip olabilir.

HTTP Rest API



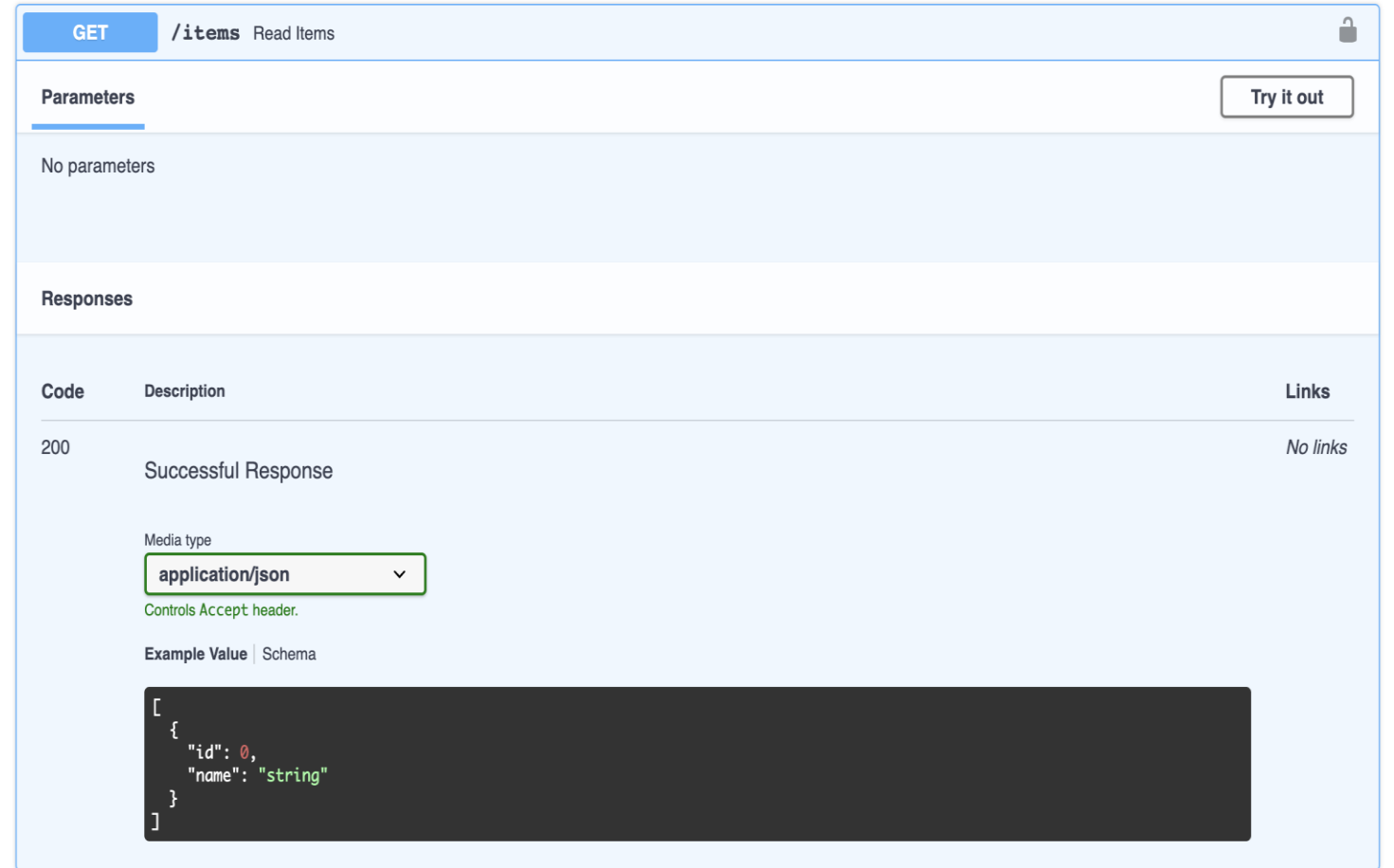
HTTP Metodları

Mikroservis



FastAPI

- FastAPI, Python'da RESTful API'ler oluşturmak için modern bir web kütüphanesidir.
- Yüksek Performans
- Kullanım Kolaylığı



The screenshot displays the Swagger UI for a REST API endpoint. At the top, a blue header bar shows the HTTP method 'GET' in a button, followed by the path '/items' and the description 'Read Items'. A lock icon is visible in the top right corner. Below the header, the 'Parameters' section is active, showing 'No parameters'. To the right of this section is a 'Try it out' button. The 'Responses' section is also active, displaying a table with one response: a 200 status code for a 'Successful Response'. To the right of this response is a 'No links' link. Below the table, there is a 'Media type' dropdown menu set to 'application/json', with a note 'Controls Accept header.' underneath. Below this, there is a link to 'Example Value' and 'Schema'. The 'Example Value' is displayed in a dark box with the following JSON structure:

```
[
  {
    "id": 0,
    "name": "string"
  }
]
```

Docker

- Uygulama konteynerleri (izole çalışma zamanı, sanal makinelerle benzer)
- Hızlı başlatma, geleneksel sanal makinelerden daha hızlı başlatılır
- Uygulamalar için temel çalışma zamanını içerir (kütüphaneler, dosyalar vb.)
 - Bizim için: randomforest sınıflandırma modelinin çalışma ortamı
- Daemon (arka plan uygulaması) olarak çalışabilir
- Resmi görüntüler (image) için Docker hub
 - Bizim için: MLFlow ve FastAPI vb.
- Özel Docker görüntüleri, Dockerfile ile oluşturulabilir
 - Bizim için: randomforest_image

Docker Temel Kavramlar

- **Konteynerler:** Uygulamaları izole bir ortamda çalıştıran, hafif ve hızlı Docker öğeleri.
- **Görüntüler (Images):** Docker konteynerlerinin çalıştırılabilir durumunu tanımlayan şablonlar.
- **Dockerfile:** Docker görüntülerini adım adım yapılandıran dosyalar.
- **Docker Compose:** Birden fazla Docker konteynerini tanımlamak ve yönetmek için kullanılan araç.
- **Docker Daemon ve Docker Client:** Docker komutlarını işleyen arka plan süreci ve kullanıcı arayüzü.

Önemli Docker Kavramları

- **Docker konteynerleri kalıcı veri saklamazlar. Konteyner durduğunda, içindeki tüm veri silinir.**
 - Çözüm: Kalıcılık için Docker konteynerlerine bağlanan ve monte edilen "volume"lar oluşturulur.
- **Docker konteynerleri doğrudan iletişim kuramazlar.**
 - Çözüm: Sanal Docker ağları oluşturulur ve sanal IP'ler atanır.
 - Bridge: Aynı Docker ana bilgisayarında birden fazla konteynerin iletişim kurabilmesi için izole Docker ağı oluşturulur.

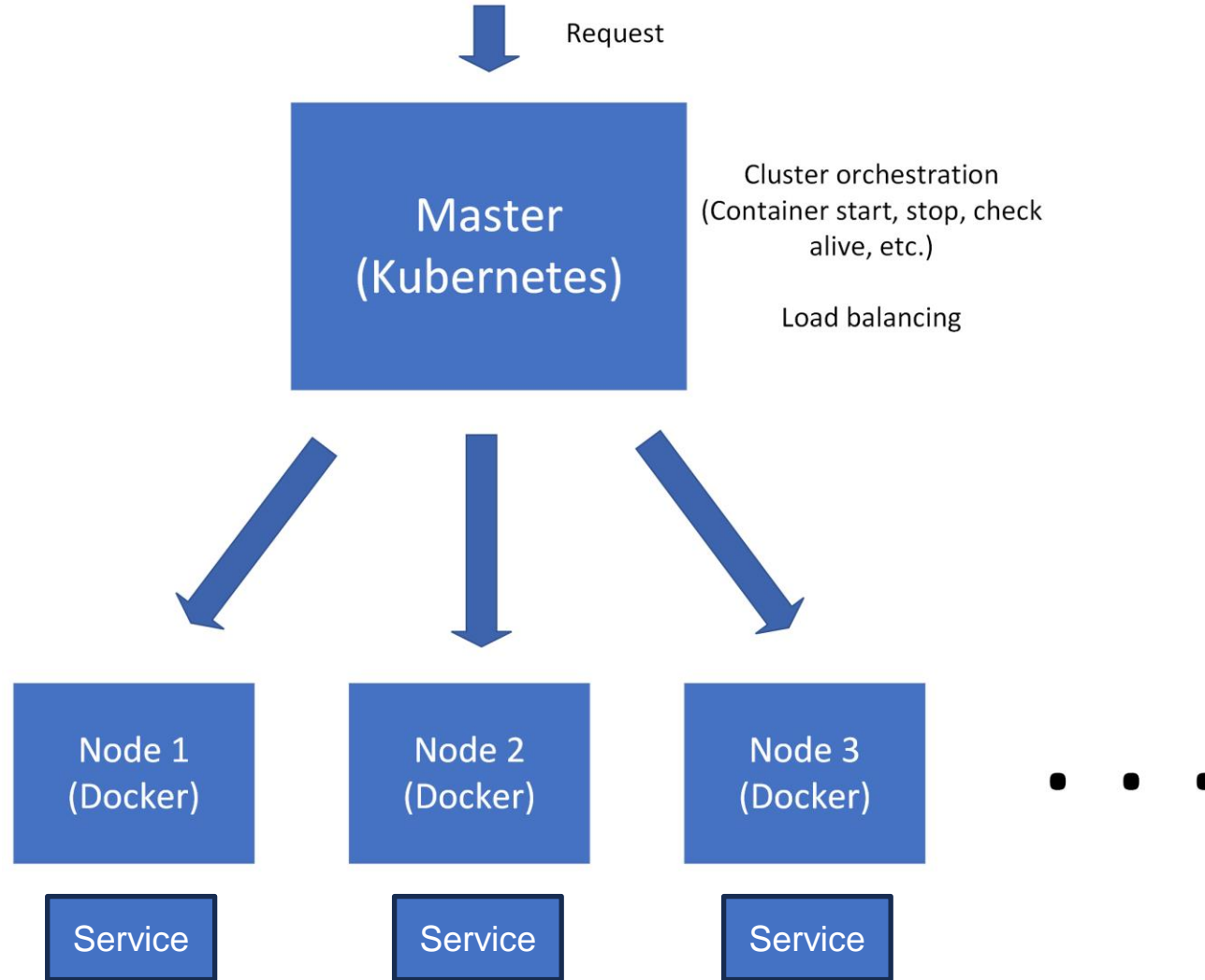
Docker Compose

- Docker Compose, hafif bir Kubernetes alternatifidir.
- Birden fazla Docker görüntüsü için orkestrasyon aracıdır.
- Docker container'larını kolayca başlatır ve durdurur.
- YAML yapılandırma dosyası, docker-compose.yml dosyasıdır.
- Yapılandırma dosyası aşağıdakileri belirler:
 - Temel Docker görüntüsü (Docker image)
 - Port bağlama ve yönlendirme (Port binding & forwarding)
 - Kalıcı Volume'lar (Persistence)
 - Sanal ağlar (Virtual networks)
 - Çalışma zamanı bağımlılıkları (Runtime dependency)
 - (diğer Docker görüntüleri)

Kubernetes

- **Mikroservisler hızlı bir şekilde karmaşık hale gelebilir, bu nedenle yönetim, orkestrasyon ve bakım gerektirir.**
- Yüksek Kullanılabilirlik (High Availability)
- Yük Dengeleme (Load Balancing)
- Ölçeklenebilirlik (Scalability)
 - Yatay Ölçeklenebilirlik (Horizontal Scalability)
- Kendini İyileştirme (Self-Healing)
 - Felaket Kurtarma (Disaster Recovery)

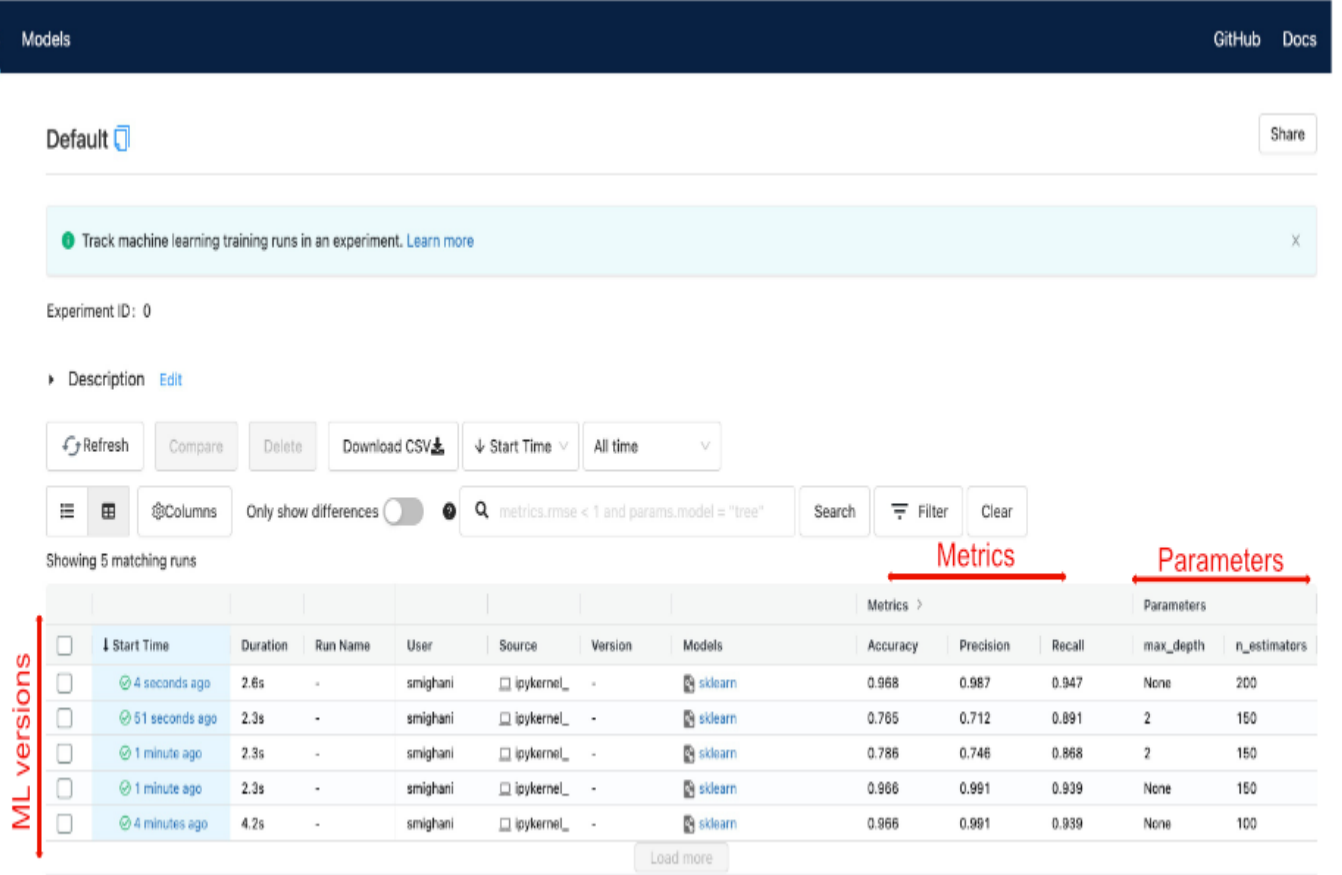
Ölçeklenebilirlik ve Yük Talebi



Horizontally scalable containers
(Add nodes as required)

MLFlow

- MLflow, makine öğrenimi projelerini izlemek, yönetmek ve tekrar üretilebilir hale getirmek için kullanılan bir platformdur.
- **İzleme ve Kayıt:** Projenin her adımını izler, kaydeder ve sonuçları görselleştirir.
- **Model Yönetimi:** Eğitilmiş modellerin kaydedilmesi, sürümlenmesi ve dağıtılması için araçlar sunar.



Models GitHub Docs

Default Share

Track machine learning training runs in an experiment. [Learn more](#)

Experiment ID: 0

Description [Edit](#)

Refresh Compare Delete Download CSV Start Time All time

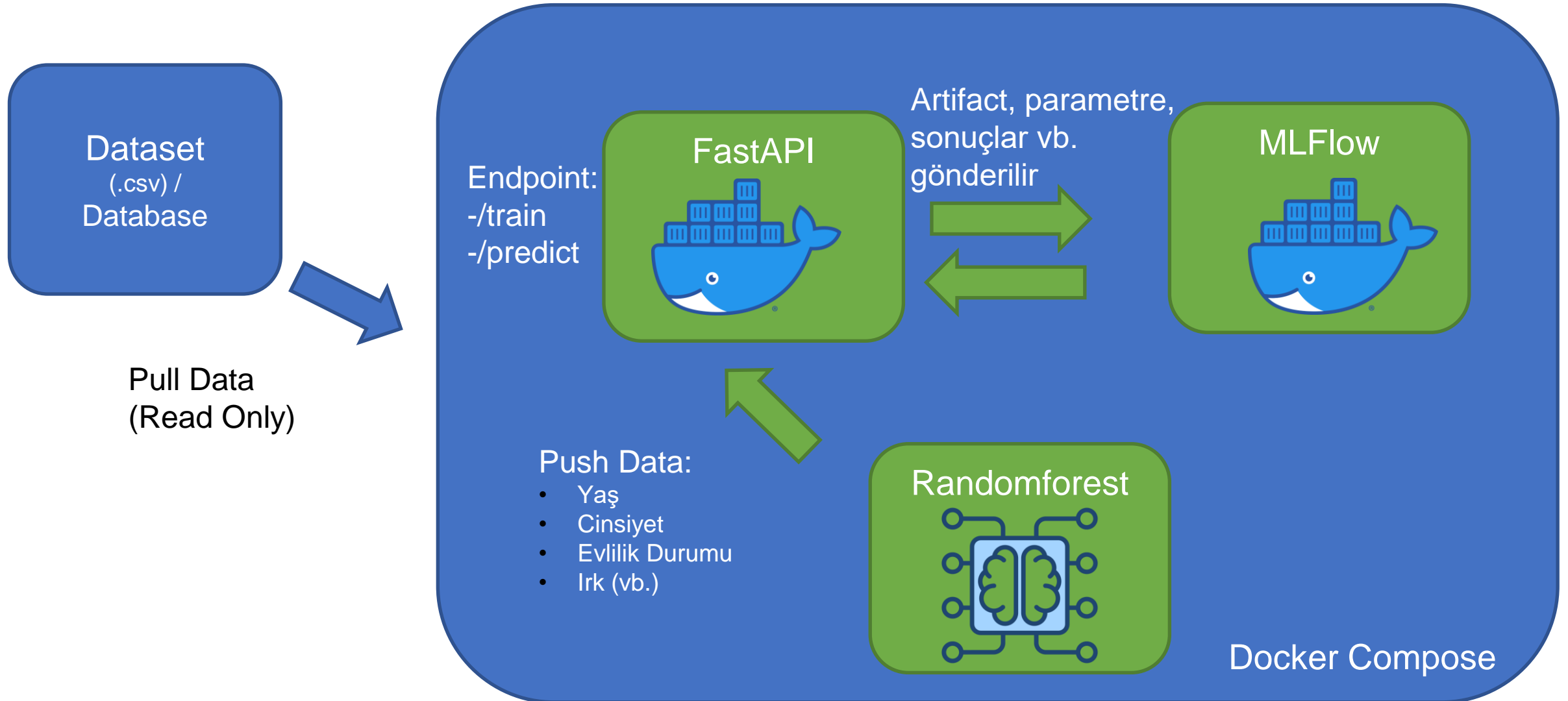
Columns Only show differences metrics.rmse < 1 and params.model = "tree" Search Filter Clear

Showing 5 matching runs

								Metrics			Parameters	
	Start Time	Duration	Run Name	User	Source	Version	Models	Accuracy	Precision	Recall	max_depth	n_estimators
	4 seconds ago	2.6s	-	smighani	ipykernel_	-	sklearn	0.968	0.987	0.947	None	200
	51 seconds ago	2.3s	-	smighani	ipykernel_	-	sklearn	0.765	0.712	0.891	2	150
	1 minute ago	2.3s	-	smighani	ipykernel_	-	sklearn	0.786	0.746	0.868	2	150
	1 minute ago	2.3s	-	smighani	ipykernel_	-	sklearn	0.966	0.991	0.939	None	150
	4 minutes ago	4.2s	-	smighani	ipykernel_	-	sklearn	0.966	0.991	0.939	None	100

Load more

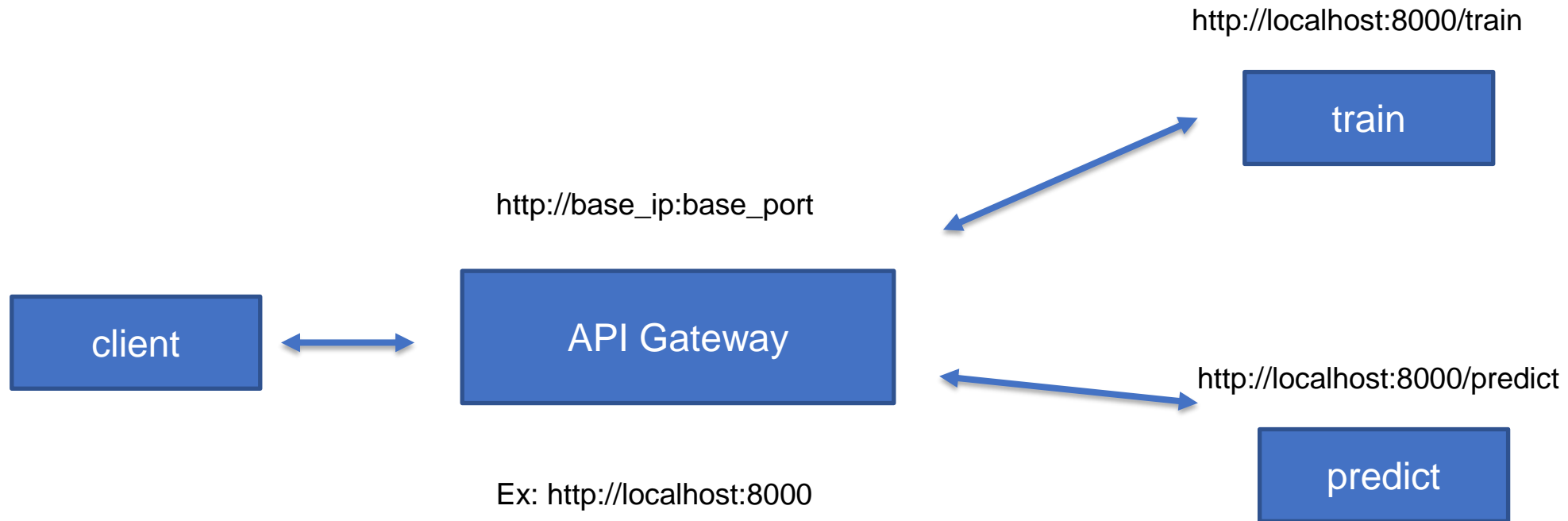
Yazılım Mimarı



Mimari Özet

- Mikroservis mimarisi
 - Her bir servis bağımsız olarak çalışan bir süreçtir
 - API aracılığıyla birbirleriyle iletişim kurabilir ve veri formatı olarak JSON kullanır
- Docker konteynerleri random forest classifier çalışma zamanıyla ilgili her şeyi içerecektir
- İşlem Akışı:
 - **Veri Çekme:** CSV dosyasından veri okunur.
 - **Eğitim:** Veriler, FastAPI uç noktasına gönderilir ve model eğitilir.
 - **Tahmin:** Veriler, FastAPI uç noktasına gönderilir ve model tahminde bulunur.
 - **Takip:** Mlfow'da model ile alakalı bütün bilgiler kaydedilir.

URL Endpoints (Mimari)



URL Endpoints

- FastAPI (geçit (gateway) olarak görev yapıyor)
- Önceden belirtilmiş URL'lere HTTP POST isteği
- İçerik formatı JSON (isteğe/yanıta)
- Desteklenen URL'ler:
 - /train
 - /predict

Genel Proje Yapısı

- **randomforest.ipynb**: Keşifsel veri analizi (EDA) için kullanılan bir Jupyter Notebook dosyası.
- **randomforest.py**: Modelinizi içeren bir Python dosyası. Random forest modelinizin tanımlandığı ve eğitildiği bir dosya.
- **app.py**: FastAPI uygulamasını çalıştıran ana dosya. Ayrıca, randomforest.py'deki fonksiyonları kullanarak eğitim ve tahmin yapar ve belirli URL'leri tanımlar.
- **dockerfile_fastapi**: FastAPI uygulamasının Docker imajını oluşturan Docker dosyası.
- **dockerfile_mlflow**: MLflow tarafından kullanılan Docker imajını oluşturan Docker dosyası.
- **requirements.txt**: app.py dosyasında kullanılan kütüphaneleri içeren bir gereksinimler dosyası.
- **Docker-compose.yaml**: MLflow ve FastAPI'nin birlikte çalıştırılması için Docker Compose dosyası. Bu dosya, her iki servisi ayağa kaldırmak ve bunları birbirine bağlamak için kullanılır.

Veri Ön İşleme

- preprocess_data:
 - '?' karakterleri yerine eksik değerler atanır.
 - Bağımsız ve bağımlı değişkenler belirlenir ve veri seti eğitim ve test kümelerine bölünür.
 - Eksik değerler, en çok görülen değerlerle doldurulur.
 - Kategorik değişkenler için One-Hot Encoding uygulanır.
 - Ölçeklendirme, RobustScaler kullanılarak gerçekleştirilir.
- Feature_importance:
 - Modelin özellik önem skorlarını belirler ve önem sırasına göre sıralanmış bir Seri döndürür.

DEMO

Teşekkürler!