KEA

Full design

Product description

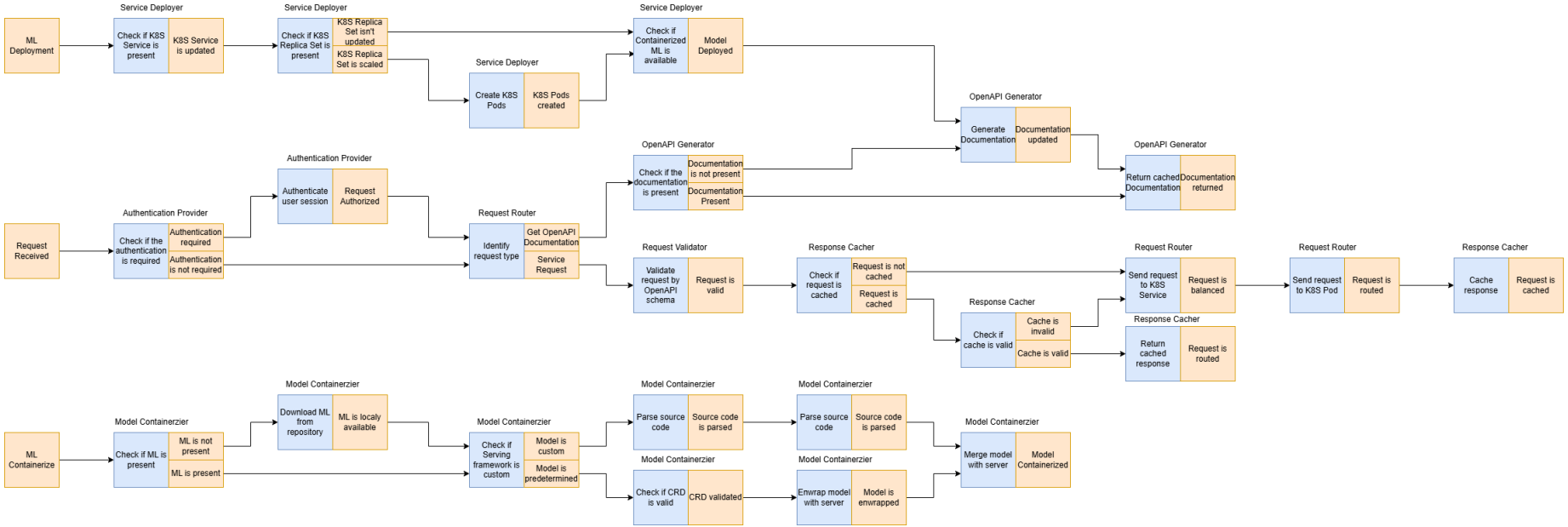
The product is a platform for deploying, managing, and scaling machine learning models in production. It offers a secure, flexible environment for automating ML tasks like model versioning, routing, and monitoring. With Kubernetes integration and containerization support, it's designed for developers, ML engineers, and enterprises needing scalable, reliable ML infrastructure.

Team K8C: Tsurkan Daniel; Dandamaev Gadji; Tsaturyan Konstantin; Smolkin Mikhail

Project repo: https://github.com/fanglores/Advanced-Software-Design

This report: https://github.com/fanglores/Advanced-Software-Design/blob/master/Practice%20Tasks/Module2/FinalTask_2/FinalTask_2.pdf

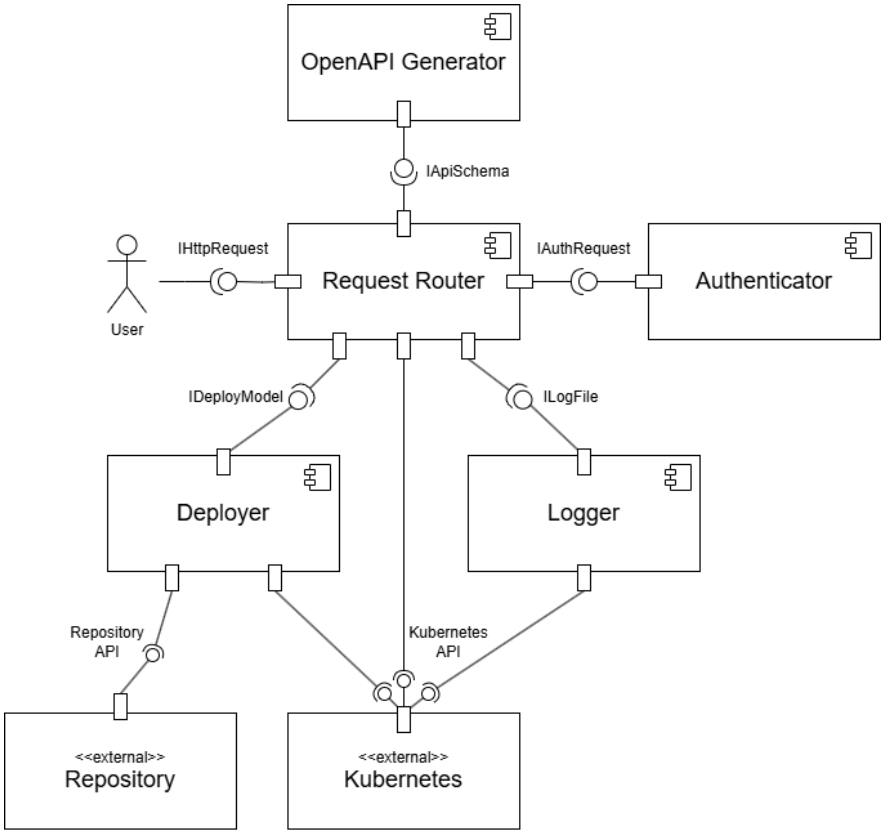
Event flow



https://github.com/fanglores/Advanced-Software-Design/tree/master/General/UseCases

System architecture

BASE
Microservices
RESTful API



Solution stack

Implementation

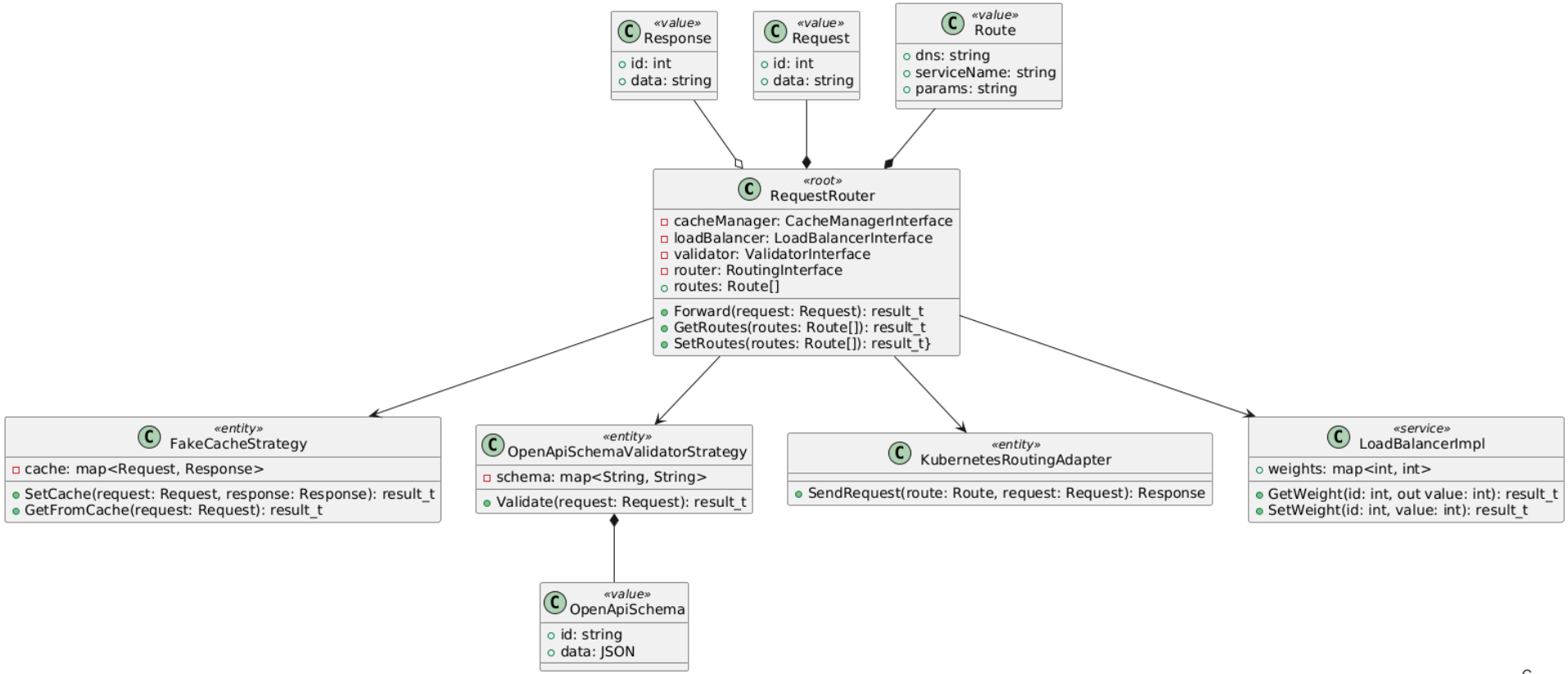
- API definition: OpenAPI
- Connection server for API: python gunicorn
- App framework: python FastAPI
- Serialization/state format: json

Testing tools pytest

Operations

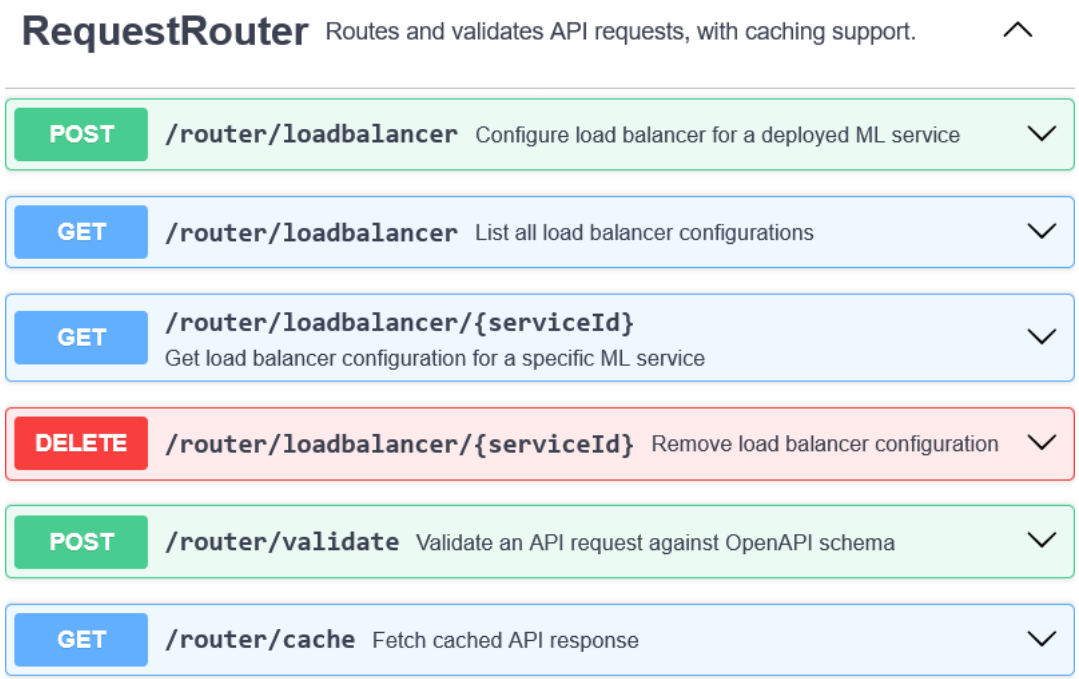
- App initializer: cookiecutter
- Code build: makefile
- CI/CD pipeline: github ci/cd
- Delivery method: docker
- Logging & monitoring: ELK

Logical data model **RequestRouter**

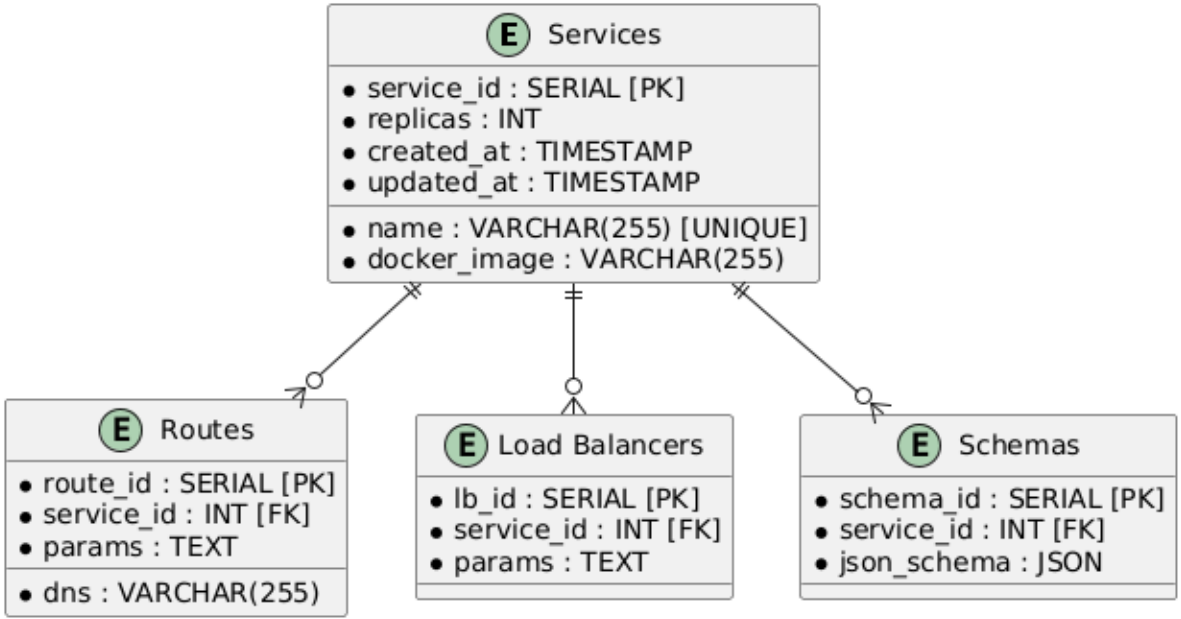


API usage **RequestRouter**

Use Case: Forward Request
Scenario:
User sends request to a service
Request is being validated by OpenAPI schema
Request is being forwarded to a specific K8s service



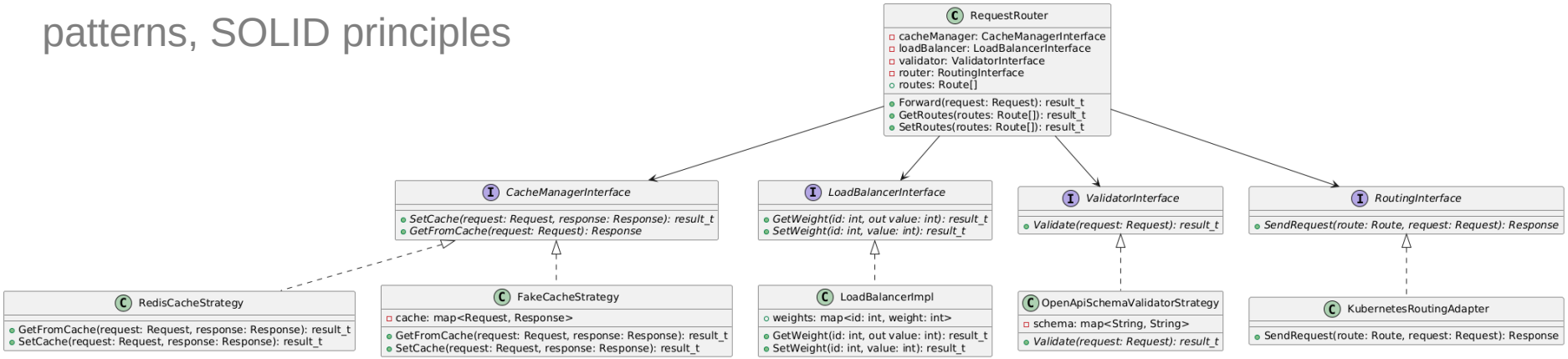
Physical schema **RequestRouter**



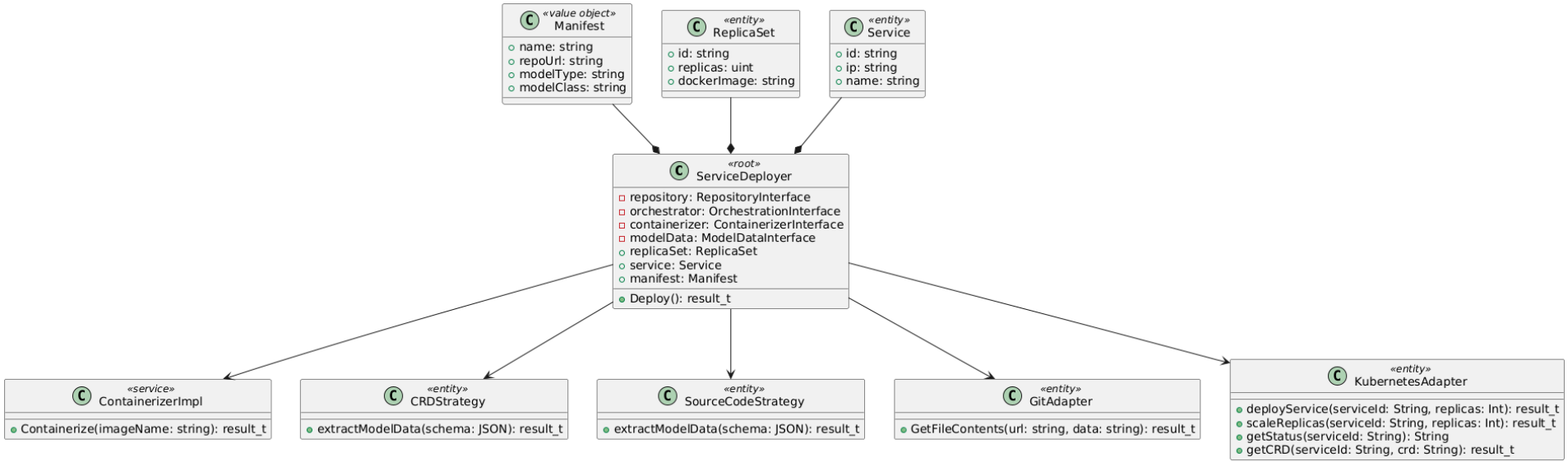
Design case for RequestRouter

Problems: strong dependency on OpenAPI schemas, cache storage, many functions

Solutions: use Strategy and Adapter patterns, SOLID principles

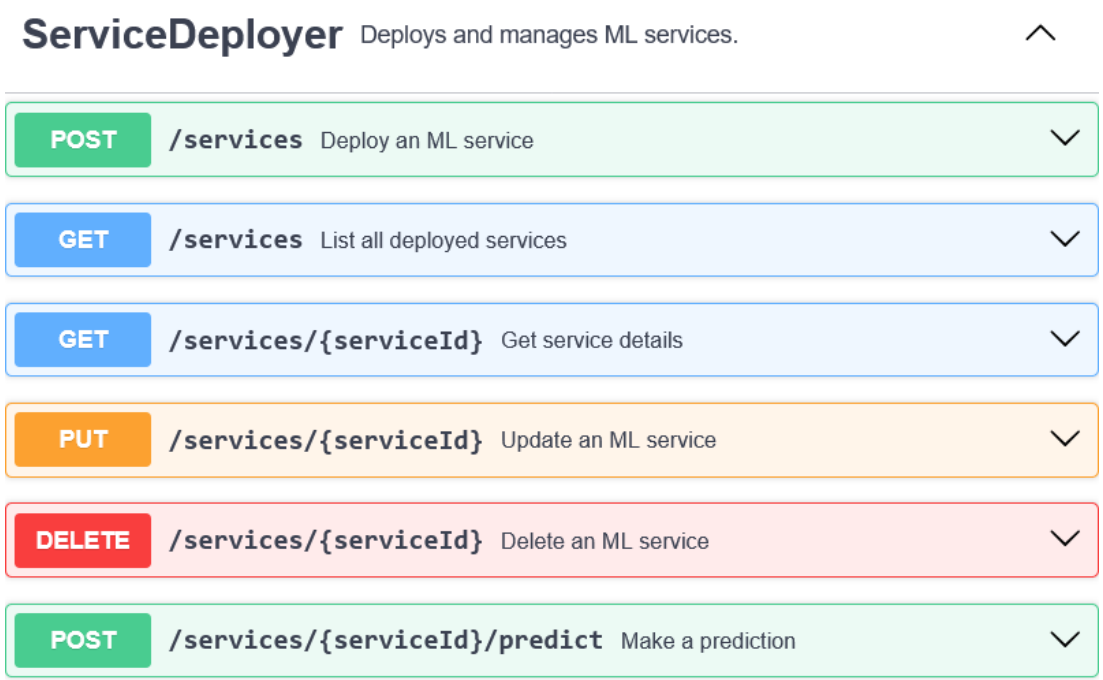


Logical data model **ServiceDeployer**

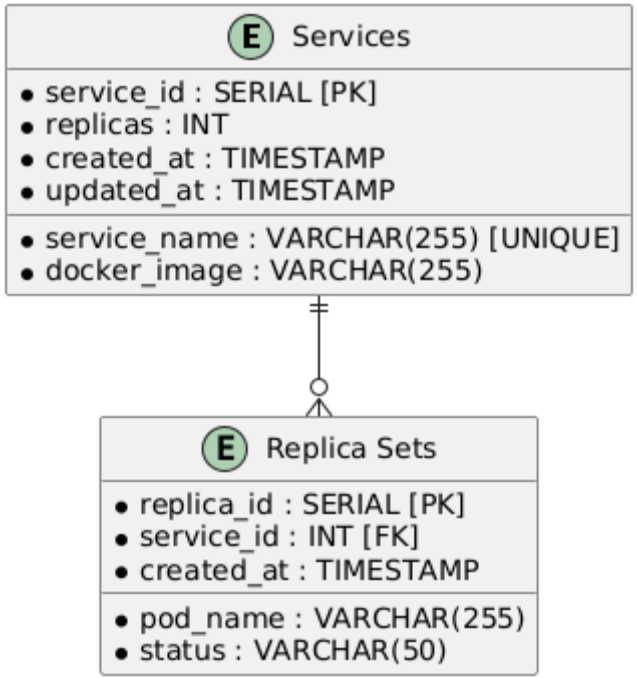


API usage **ServiceDeployer**

Use Case: Deploy Service
Scenario:
User sends request to deploy ML Model
ML Wrappers creates docker container
Docker container is deployed via service into K8s



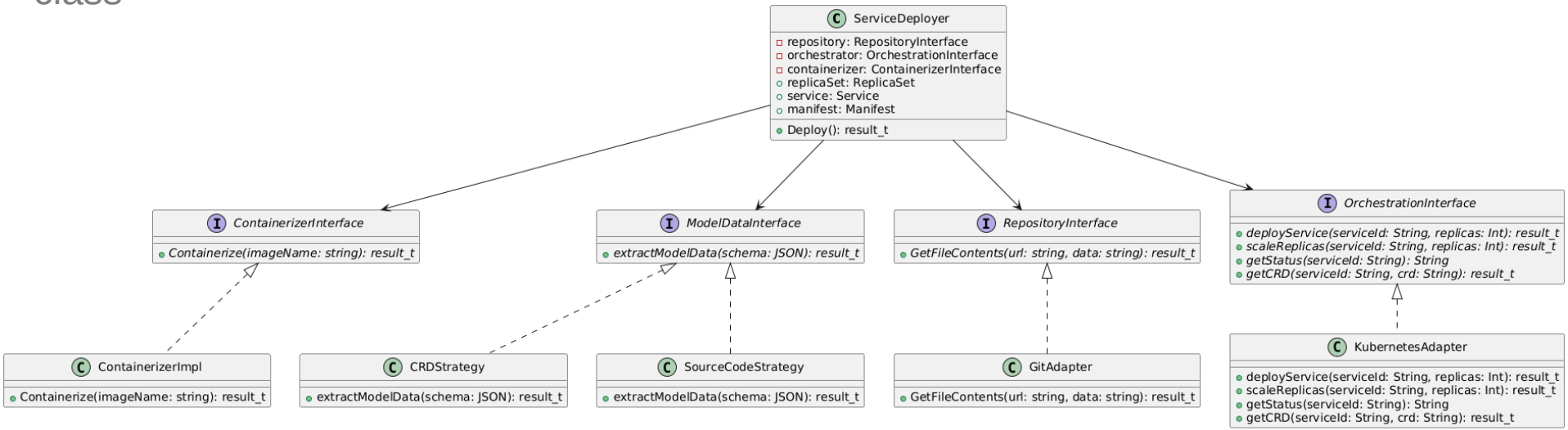
Physical schema **ServiceDeployer**



Design case of Service Deployer

Problems: new deploy strategies require changes in ServiceDeployer; ServiceDeployer can work with different data, many functions in one class

Solutions: use SOLID principles, Adapter and Strategy patterns



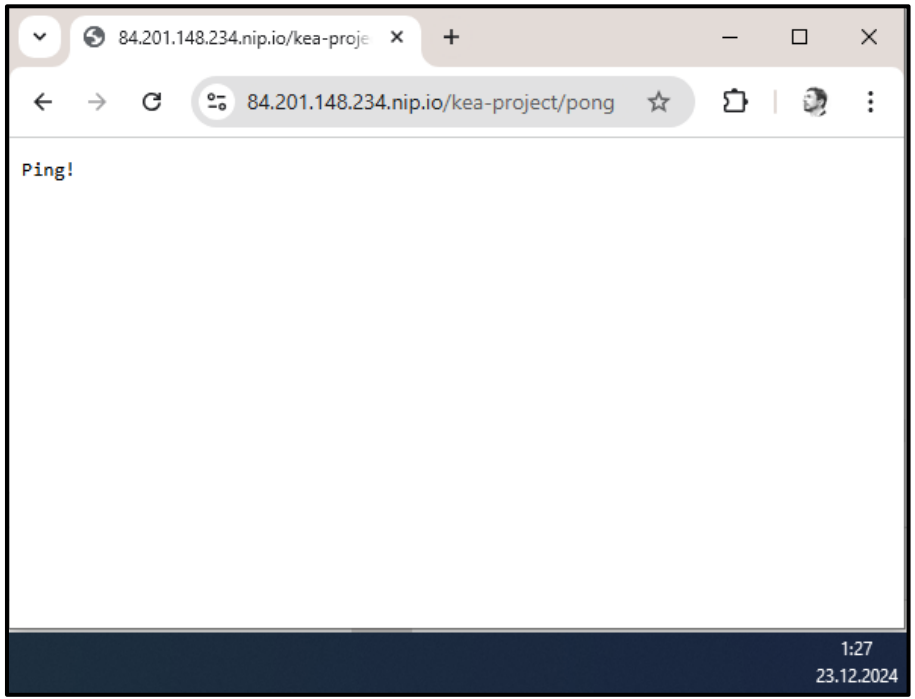
Design complexity: Service Deployer

Class/Metrics	WMC	CBO	NOC	DIT
ServiceDeployer	1	8	0	1
ContainerizerImpl	1	1	0	2
CRDStrategy	1	1	0	2
SourceCodeStrategy	1	1	0	2
GitAdapter	1	1	0	2
KubernetesAdapter	4	1	0	2

Design complexity

Service	SIY	AIS	ADS
Request Router	0	4	0
Authenticator	0	0	1
OpenAPI Generator	0	0	1
Deployer	0	0	1
Logger	0	0	1
AVG	-	0,8	0,8

System demo (TBD)



http://84.201.148.234.nip.io/kea-project/pong
http://84.201.148.234.nip.io/kea-project/ping

Repository structure

Repository structure

```
.
├── product_img.jpg
├── README.md
├── General/
│   ├── Domain_description_en.md
│   ├── Task_description_en.md
│   ├── DFD0/
│   ├── StoryMap/
│   └── UseCases/
└── Practice Tasks/module2
    ├── FinalTask_2/
    ├── Task_8/
    ├── Task_9/
    ├── Task_9.1/
    ├── Task_10/
    ├── Task_11/
    └── Task_12/
```

Tools Used:

- Github
- Drawio
- Planttext
- Swagger

Team and roles

Class diagrams, design complexity	Use cases, design cases	Design cases/patterns, logical and physical schemas, components diagram	Event flow, API definition, K8s deployment, components diagram
Tsurkan Daniel Tg: @crazy_deyzi	Dandamaev Gadji Tg: @dandamaev	Tsaturyan Konstantin Tg: @fanglores	Smolkin Mikhail Tg: @m0hnatik