

# K8s Nemesis

API Design

# Product description

A cloud-native service designed to enhance application scalability and performance in Kubernetes (K8s) environments, specifically targeting machine learning (ML) applications with unique scaling and resource requirements.

Team: Sergey Lokhmatikov, Roman Kuzmenko, Andrey Tamplon

Repo: [https://github.com/Lokhmat/k8s\\_nemesis/tree/main](https://github.com/Lokhmat/k8s_nemesis/tree/main)

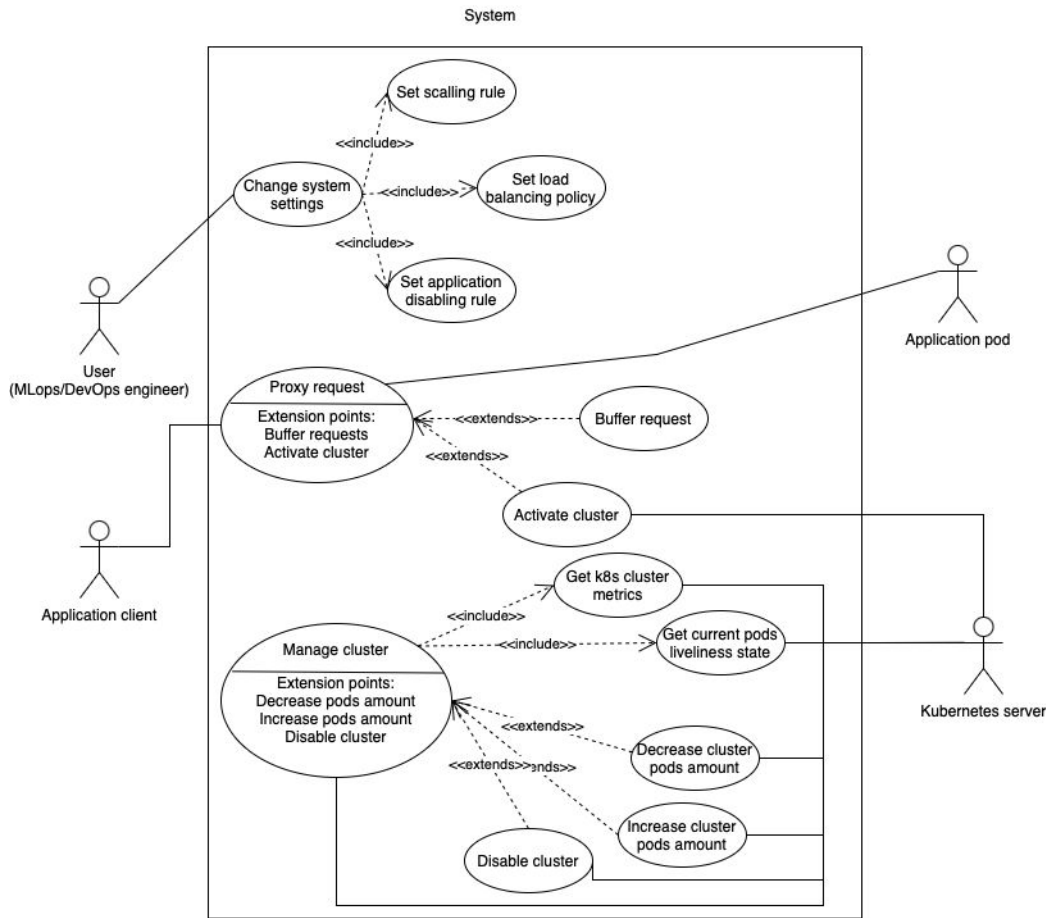
Link to API: [https://github.com/Lokhmat/k8s\\_nemesis/tree/main/api](https://github.com/Lokhmat/k8s_nemesis/tree/main/api)

Report: [https://github.com/Lokhmat/k8s\\_nemesis/blob/main/task\\_10\\_slides.pdf](https://github.com/Lokhmat/k8s_nemesis/blob/main/task_10_slides.pdf)

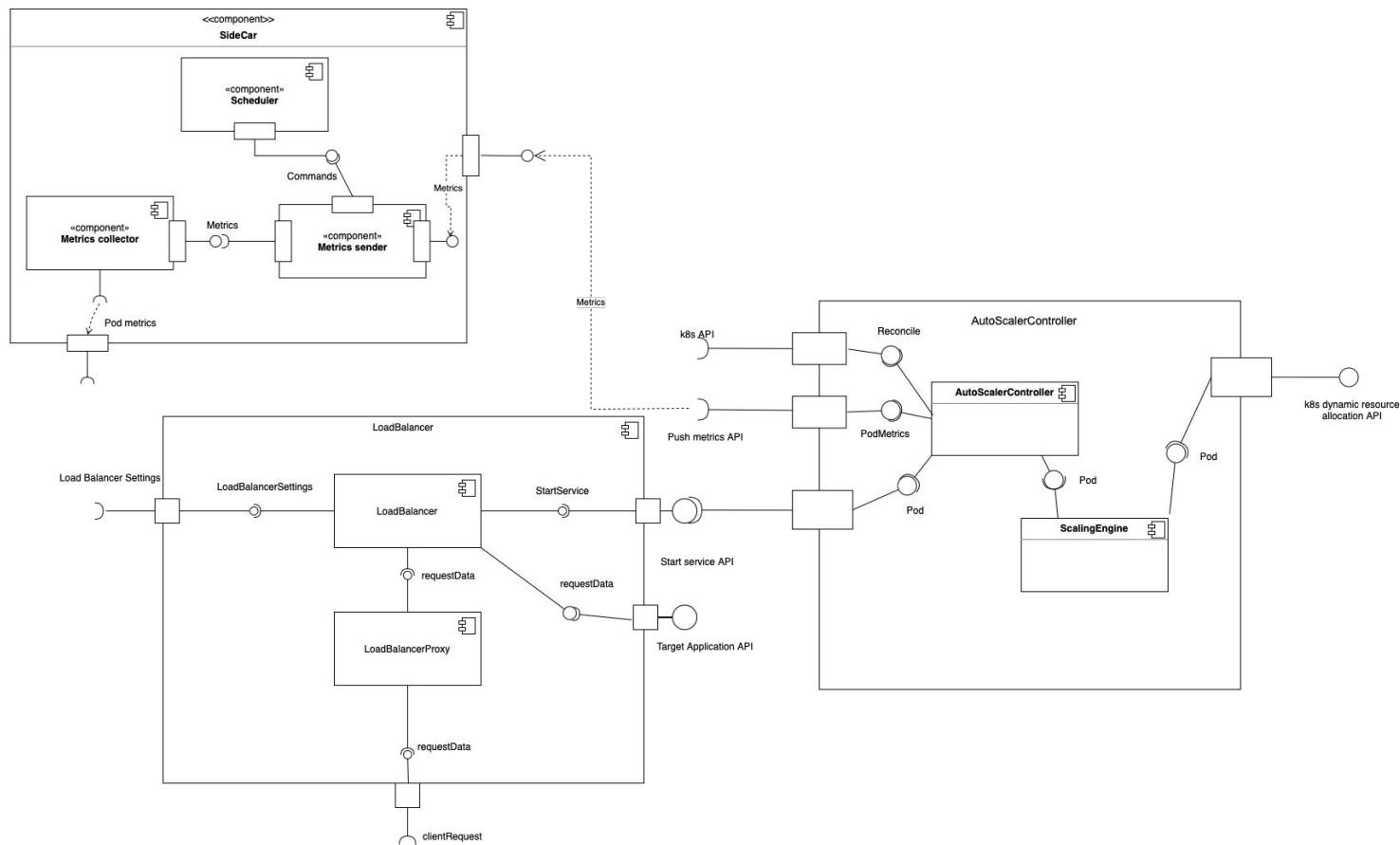
# Use case diagram or event flow

Textual use case scenarios:

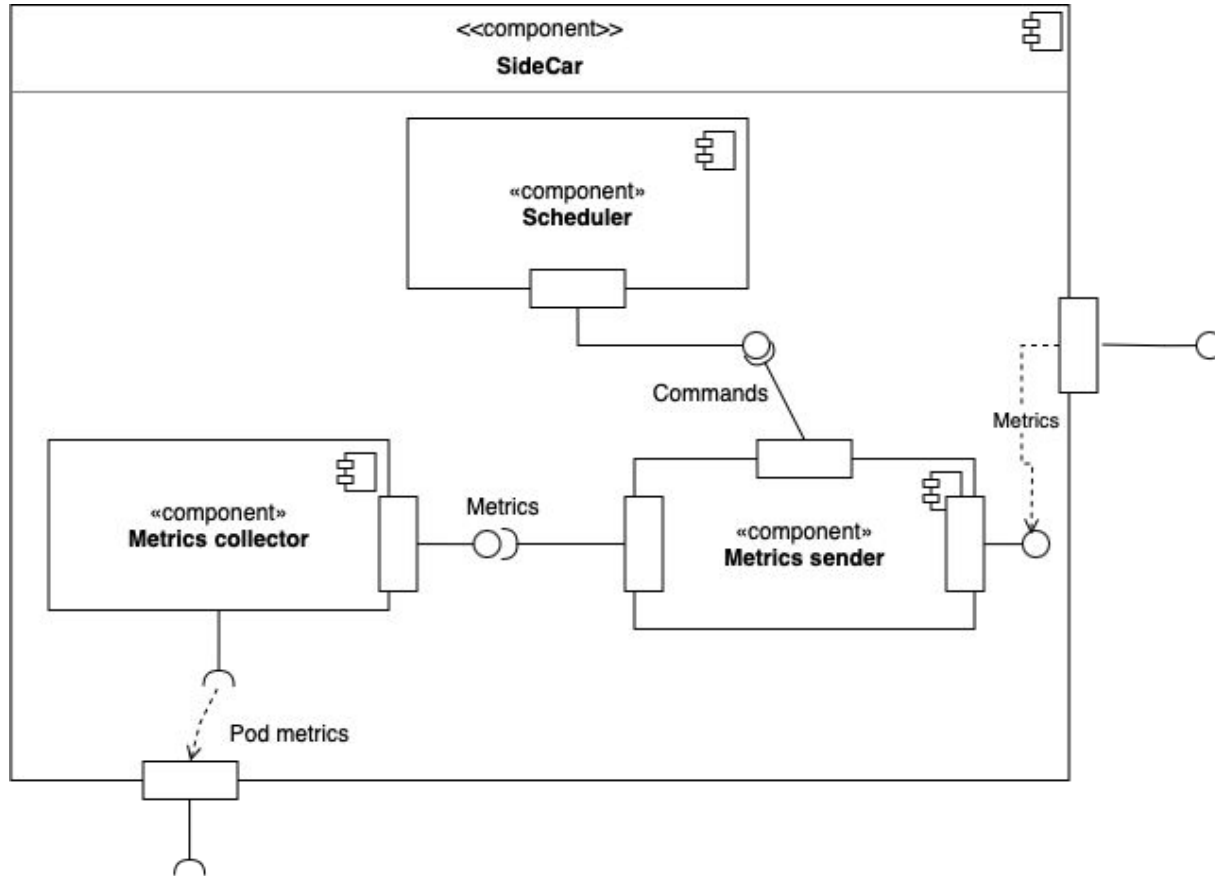
[https://github.com/Lokhmat/k8s\\_nemesis/blob/main/final\\_task\\_materials/textual\\_use\\_cases.md](https://github.com/Lokhmat/k8s_nemesis/blob/main/final_task_materials/textual_use_cases.md)



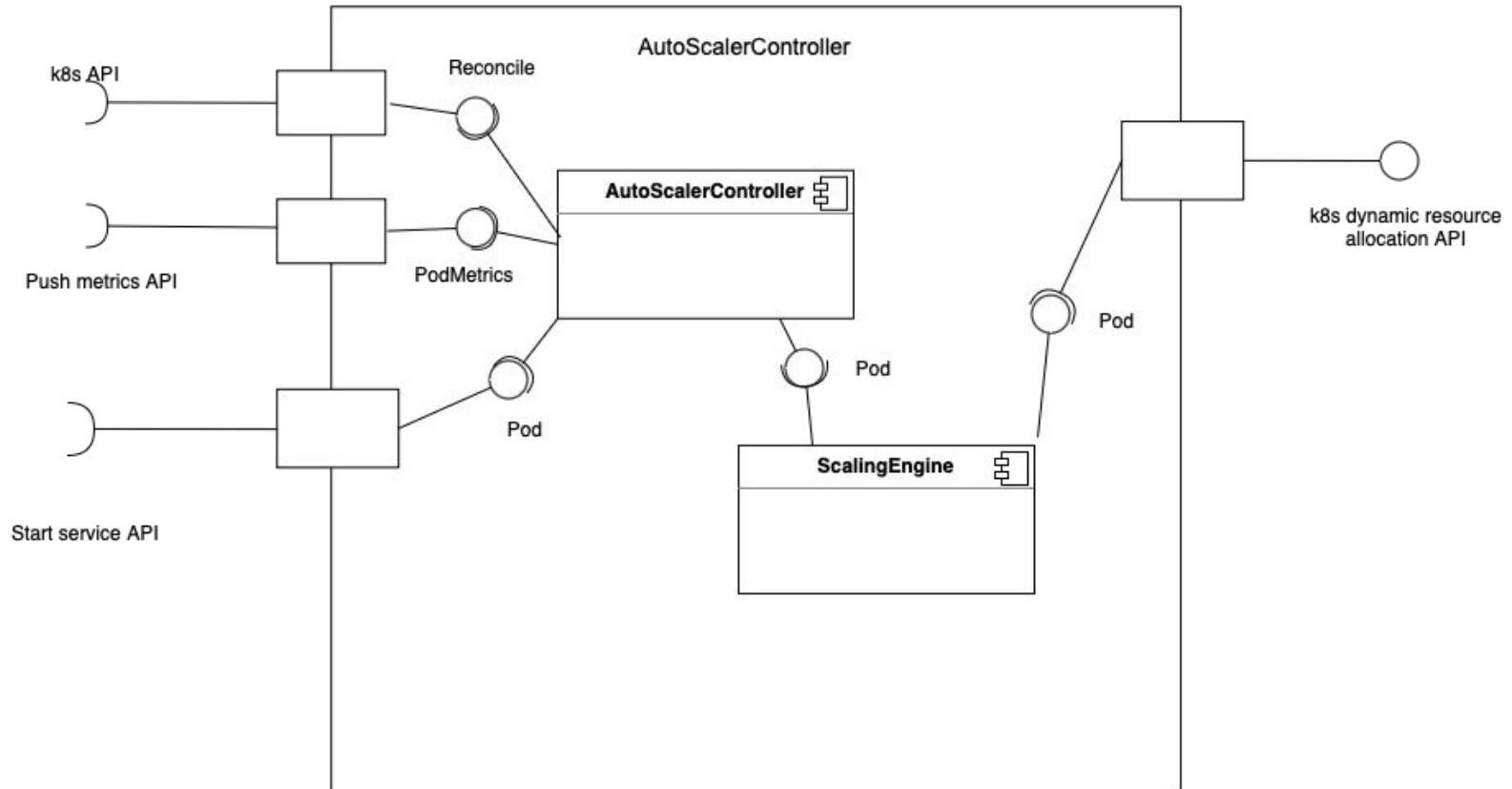
# Service diagram



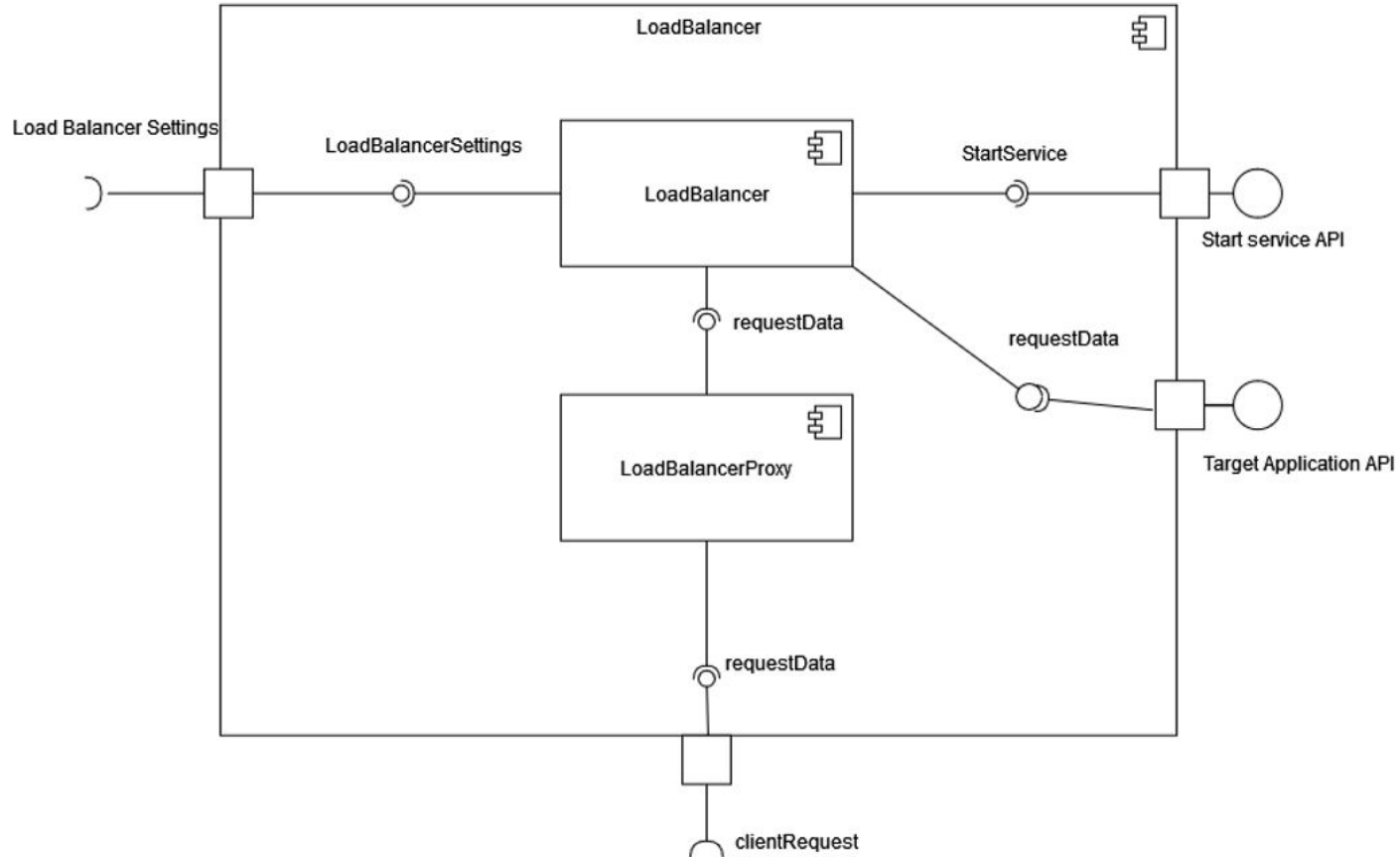
# Service diagram - SideCar



# Service diagram - AutoScalerController



# Service diagram - LoadBalancer



# API usage Autoscaler

Steps for accomplishing these use cases - Set application disabling rule, Set scaling rule, Set load balancing rule:

1. List CRs - first handler in list [GET for CRs in namespace]
2. If there is a CR for the desired deployment - update it (set needed rule) [PATCH or PUT]
3. Else create it [POST]
4. Now you can see it change by [GET by name]

GET	<code>/apis/autoscaler-group.autoscaler/v1alpha1/namespaces/{namespace}/customautoscalers</code> Получить список всех CustomAutoscaler ресурсов в указанном пространстве имен	✓
POST	<code>/apis/autoscaler-group.autoscaler/v1alpha1/namespaces/{namespace}/customautoscalers</code> Создать новый CustomAutoscaler ресурс	✓
GET	<code>/apis/autoscaler-group.autoscaler/v1alpha1/namespaces/{namespace}/customautoscalers/{name}</code> Получить конкретный CustomAutoscaler ресурс	✓
PUT	<code>/apis/autoscaler-group.autoscaler/v1alpha1/namespaces/{namespace}/customautoscalers/{name}</code> Обновить существующий CustomAutoscaler ресурс	✓
PATCH	<code>/apis/autoscaler-group.autoscaler/v1alpha1/namespaces/{namespace}/customautoscalers/{name}</code> Частично обновить существующий CustomAutoscaler ресурс	✓



# API usage Autoscaler

This is api for “Get k8s cluster metrics” and “Get k8s pods liveness state” use cases in which sidecars use this handler to send metrics to autoscaler:

1. Sidecar collects metrics and calls /metrics [POST]

**POST**

**/metrics** Получить метрики от сайдкара в автоскейлер



# API usage LoadBalancer

## 1. Proxy request

- GET Get logs of proxied requests
- POST Proxy a request to a target service
- POST Send a request directly to the Target Application API
- GET Retrieve data from the Target Application API
- PATCH Partially update data in the Target Application API
- PUT Update data in the Target Application API

## 2. Set Load balancing policy

- GET Get current load balancing settings
- PUT Update load balancing settings

GET	/loadbalancer/settings	Get current load balancing settings	▼
PUT	/loadbalancer/settings	Update load balancing settings	▼
POST	/proxy/request	Proxy a client request to a target service	▼
GET	/proxy/logs	Get logs of client requests	▼
GET	/target-app/api	Retrieve data from the Target Application API	▼
POST	/target-app/api	Send a request to the Target Application API	▼
PUT	/target-app/api	Update data in the Target Application API	▼
PATCH	/target-app/api	Partially update data in the Target Application API	▼

# Solution stack

## Implementation

- API definition: OpenAPI
- Connection server for API: go standard library
- App framework: go standard library
- Serialization/state format: json

## Testing tools: pytest

## Operations

- App initializer: -
- Code build: Makefile
- CI/CD pipeline gitlab: CI/CD
- Delivery method: Kubernetes package
- Logging & monitoring: ELK stack(either one already available in project or new one specifically for load balancer)