Conductor

Универсальный оркестратор бизнес-процессов **Conductor**. Обзор возможностей и их расширение.

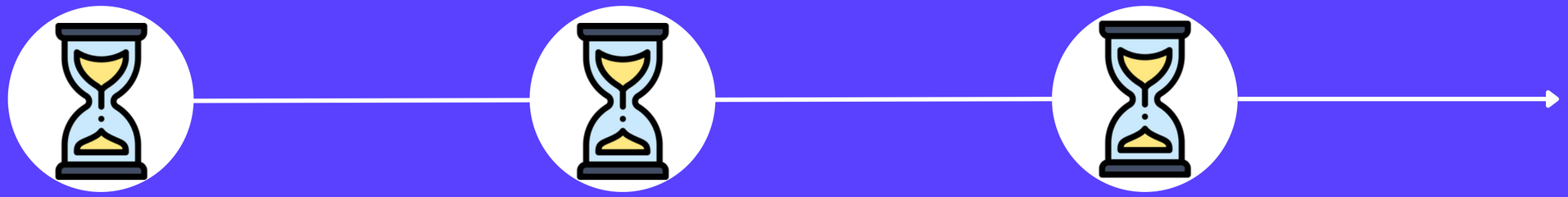
Королькевич Глеб
Начальник отдела разработки

О ЧЕМ ДОКЛАД

- Альтернативы
- Обзор Conductor
- Реальные примеры
- Опыт эксплуатации

Обо мне

Роли при работе с Conductor



Разработчик Workflow

Тестировщик Workflow

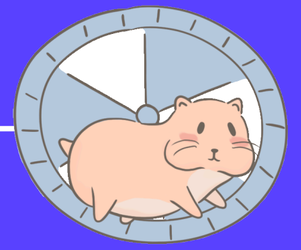
Сопровождение Workflow

Обо мне

Роли при работе с Conductor

Разработчик платформы

Техлид Платформы



IT manager

Camunda

CAMUNDA

zeebe

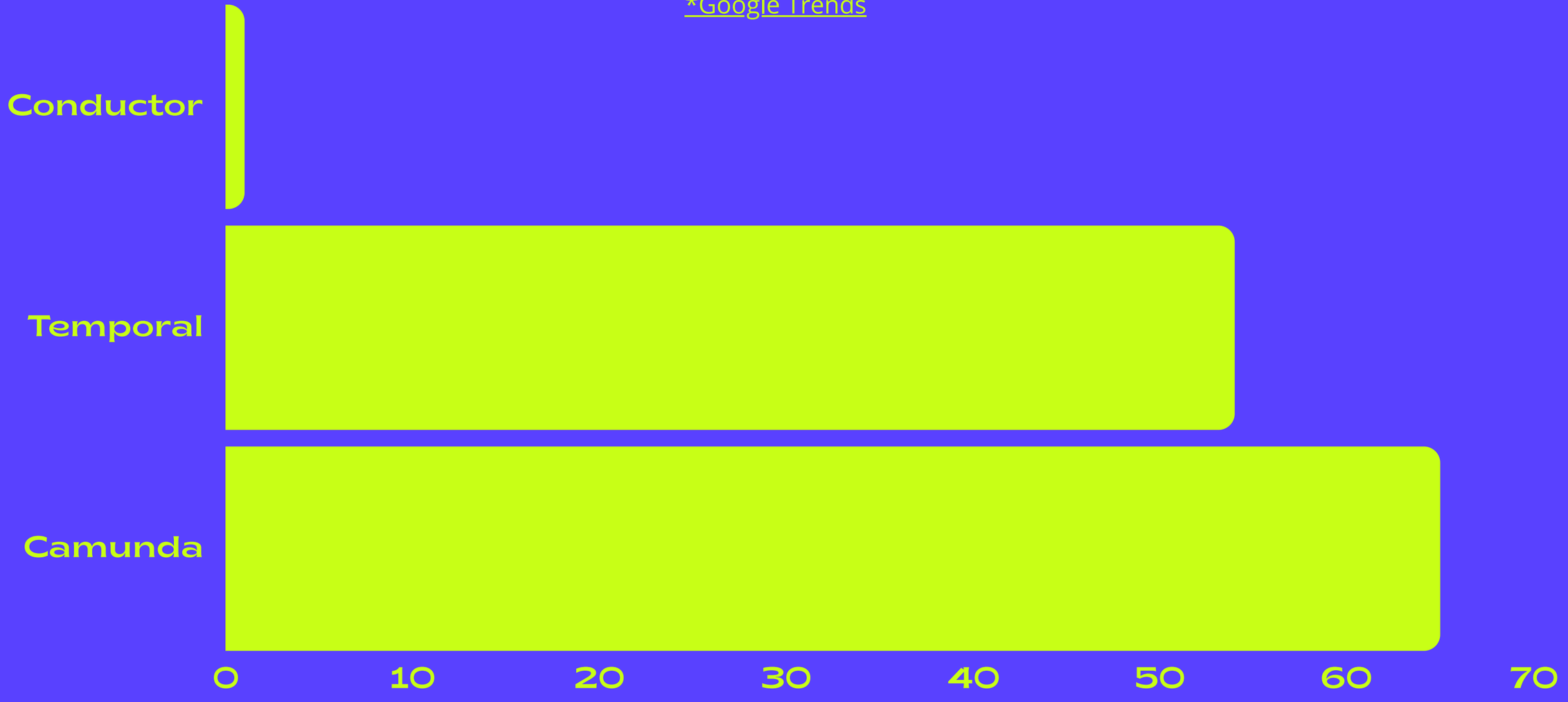
by Camunda

Temporal (ex. Cadence)

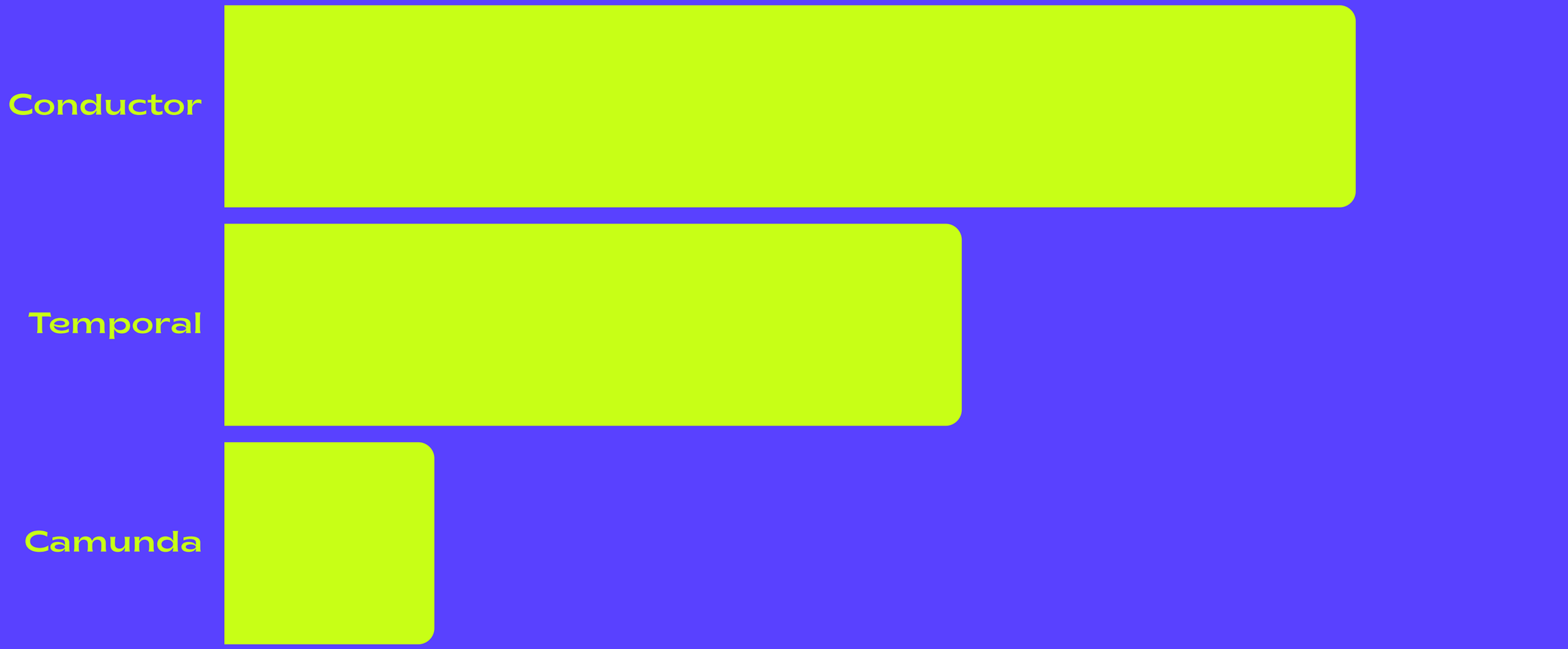
Conductor

Популярность в РФ

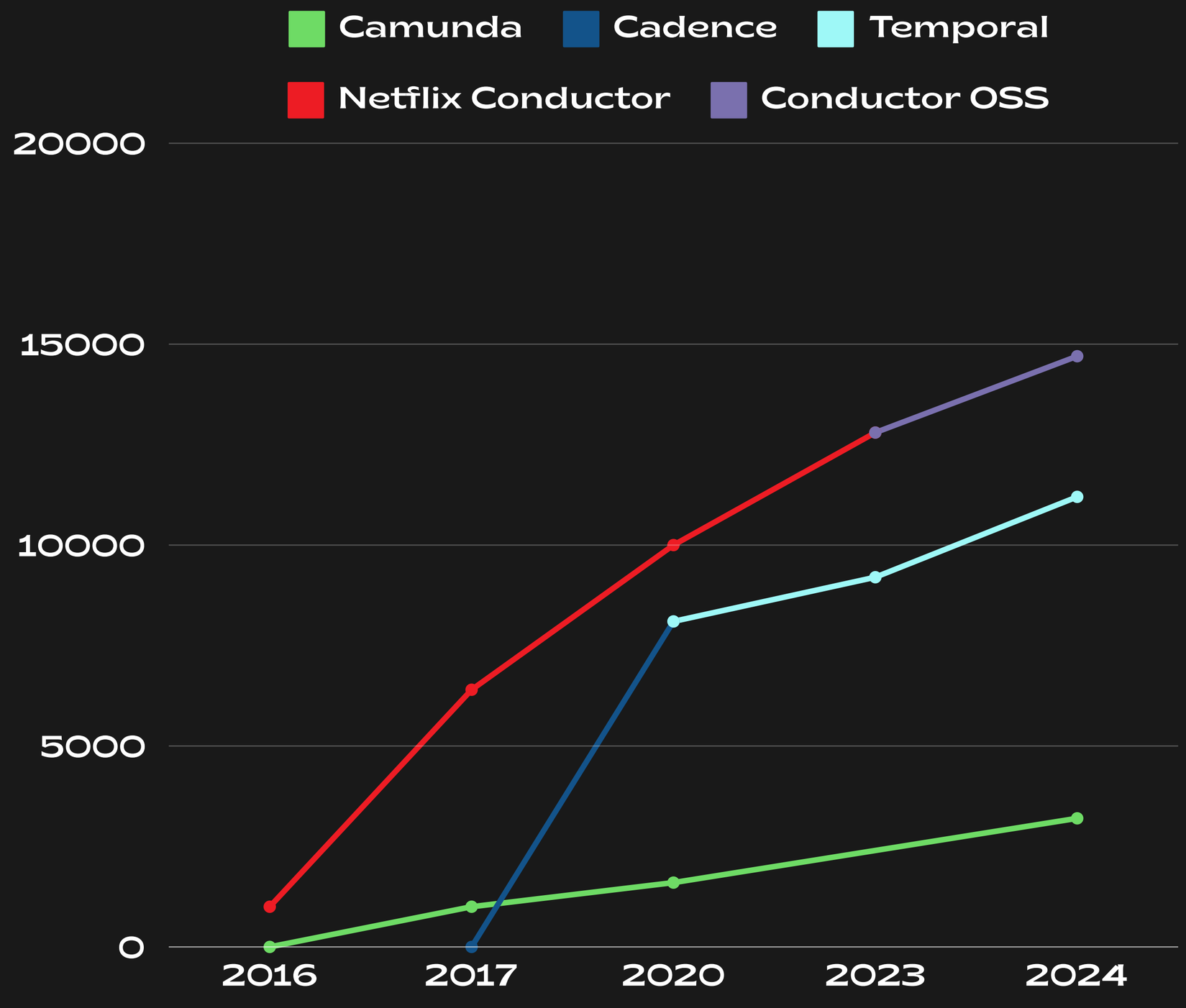
*Google Trends



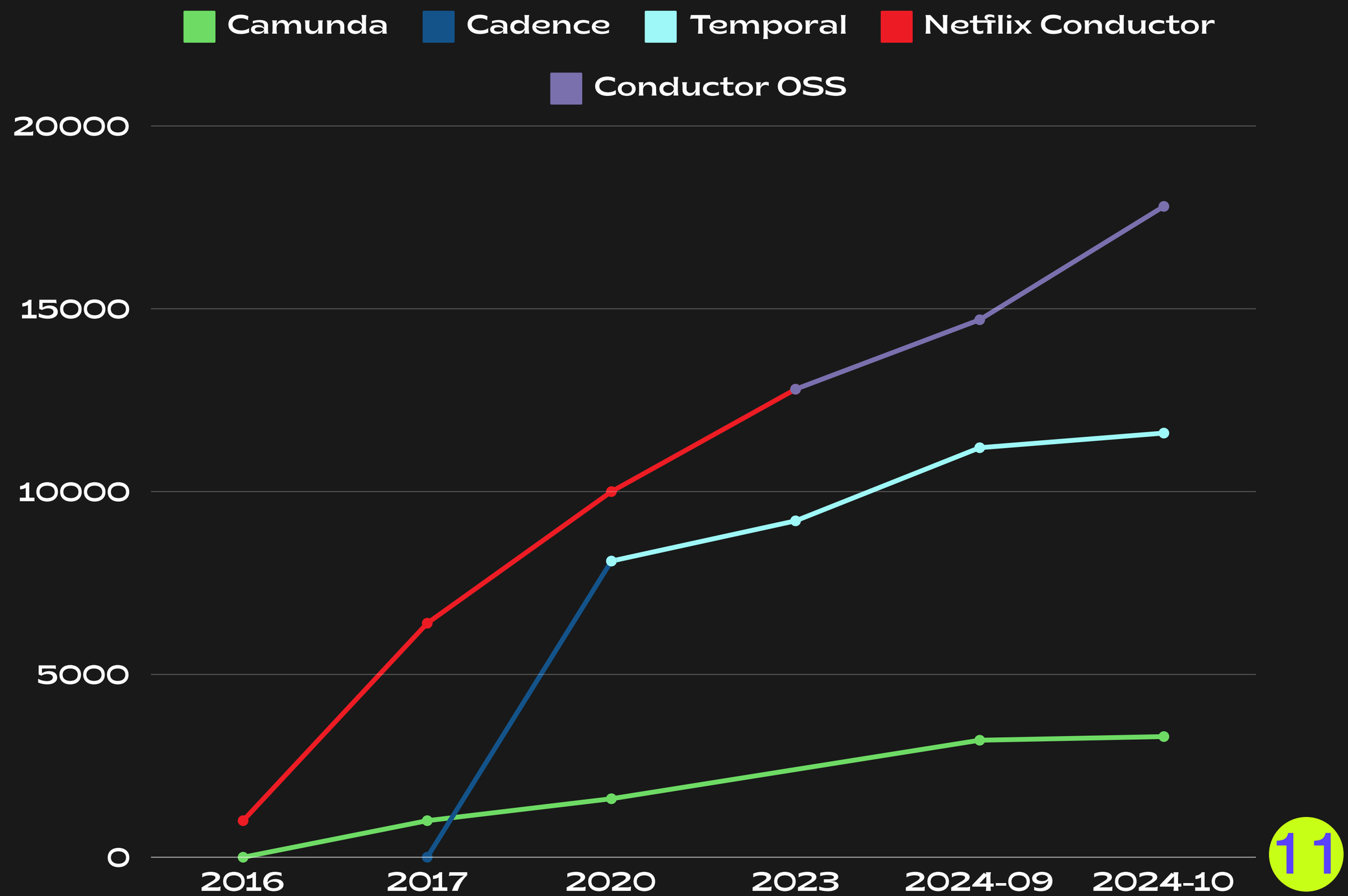
Популярность в мире



Github stars



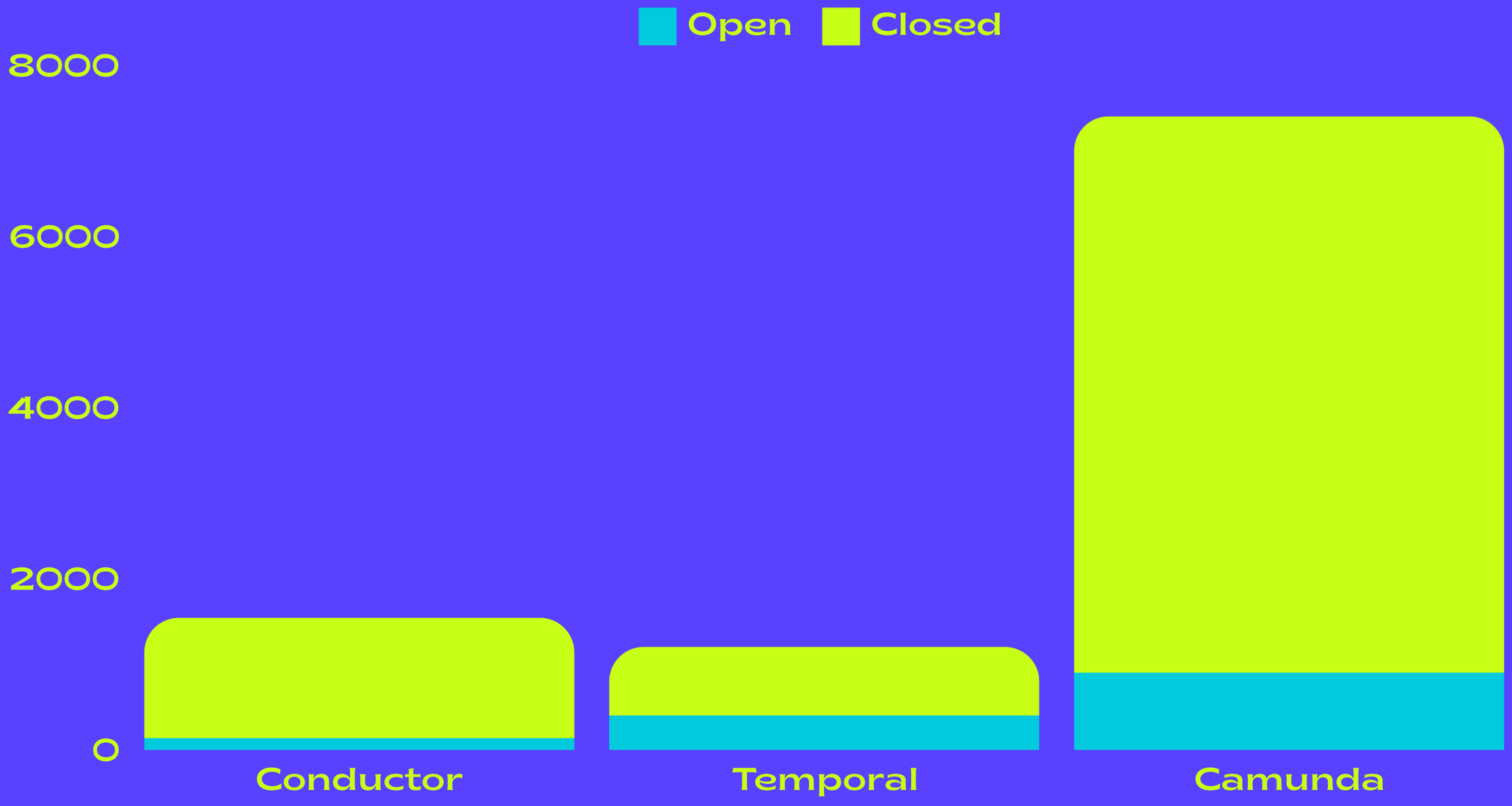
Github stars



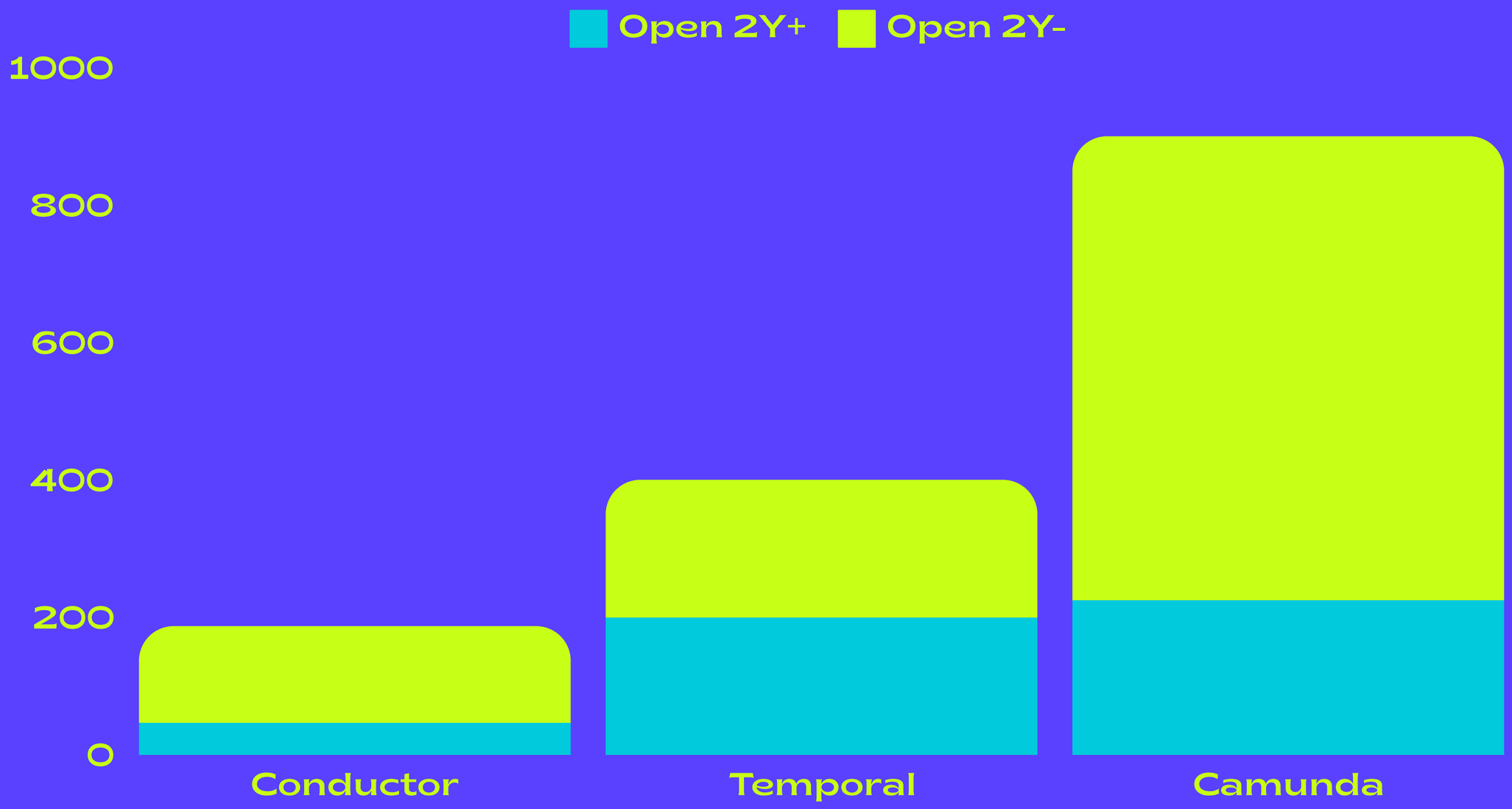
Github issues

Issues\Name	Temporal	Camunda	Conductor
OPEN	400	900	140
CLOSED	800	6500	1400
OPEN %	33%	12%	9%
OPEN 2 years+	17%	3%	3%

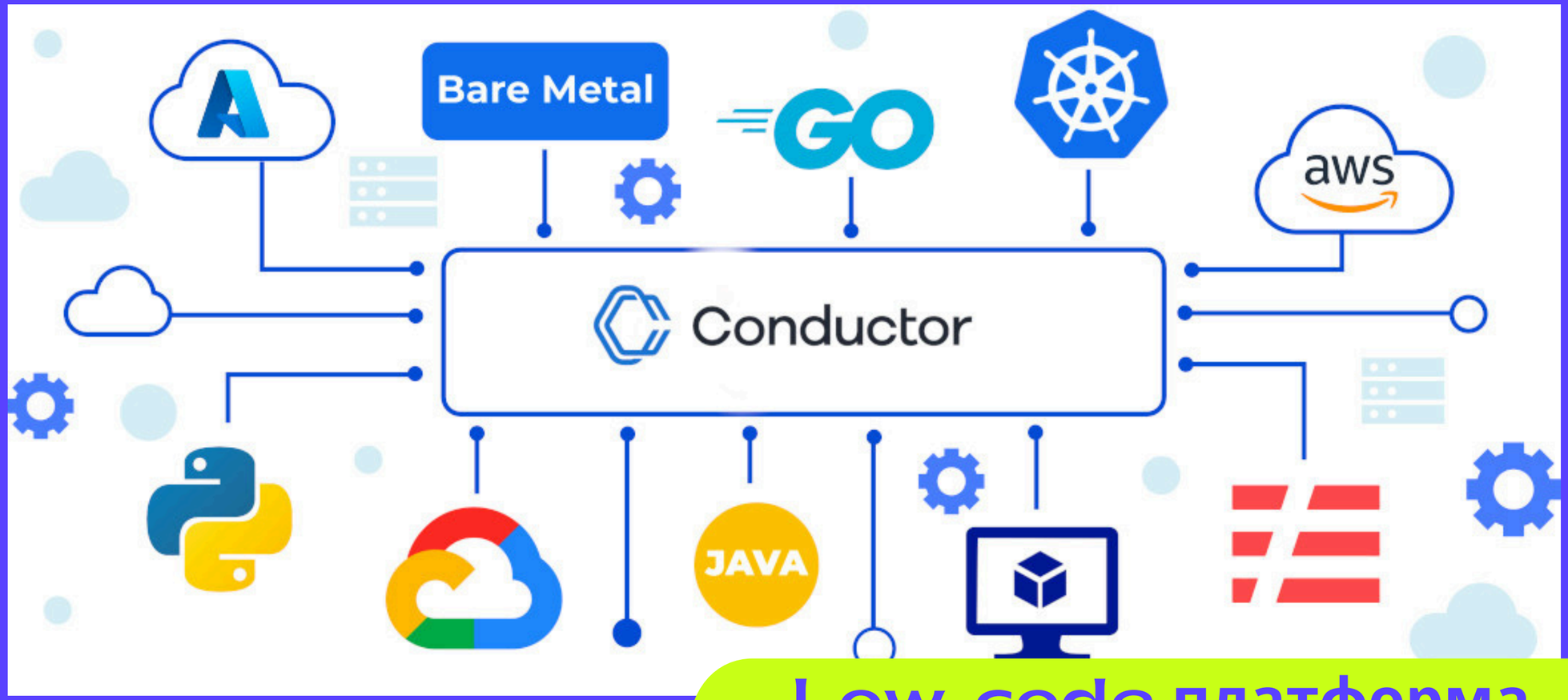
Github issues



Github open issues

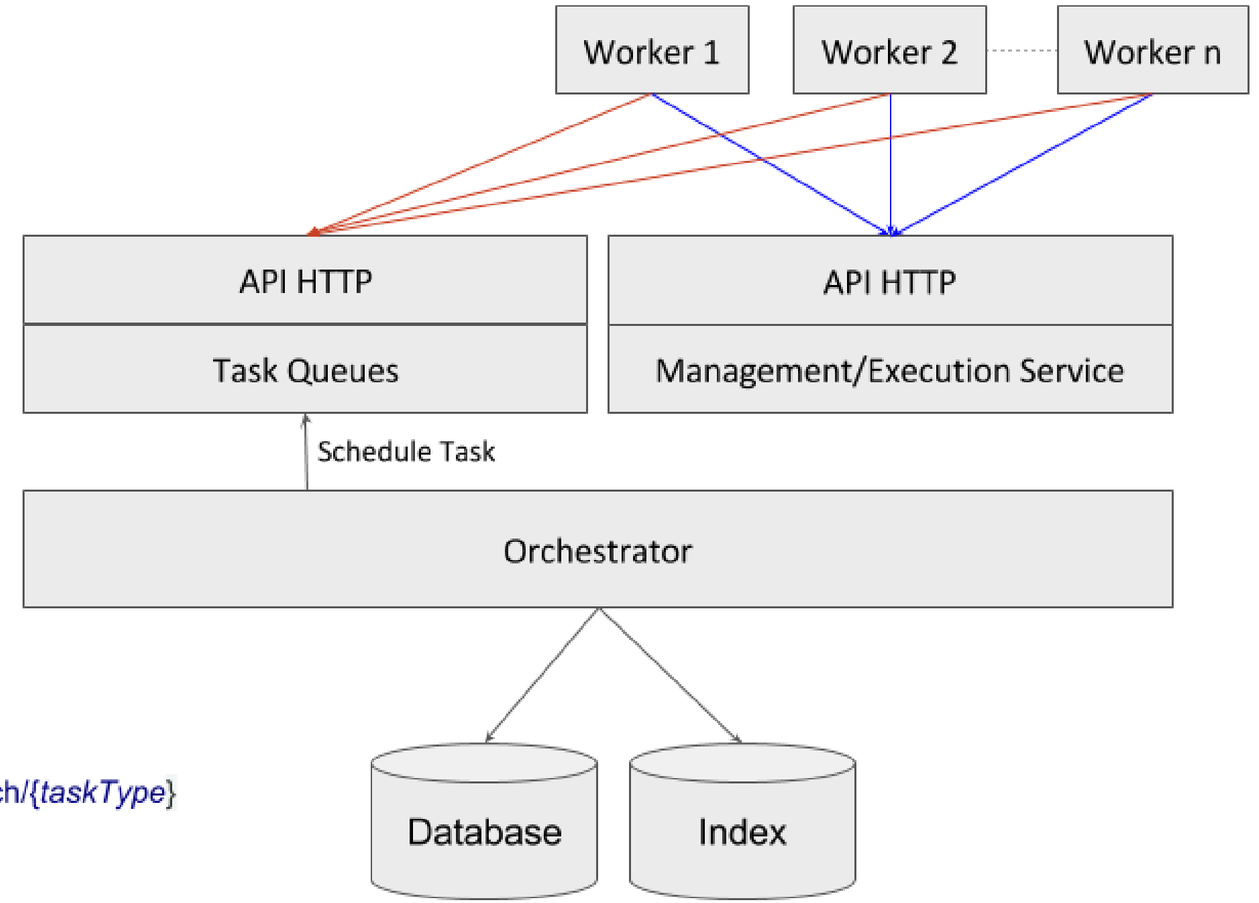


Conductor



Low-code платформа бизнес оркестрации

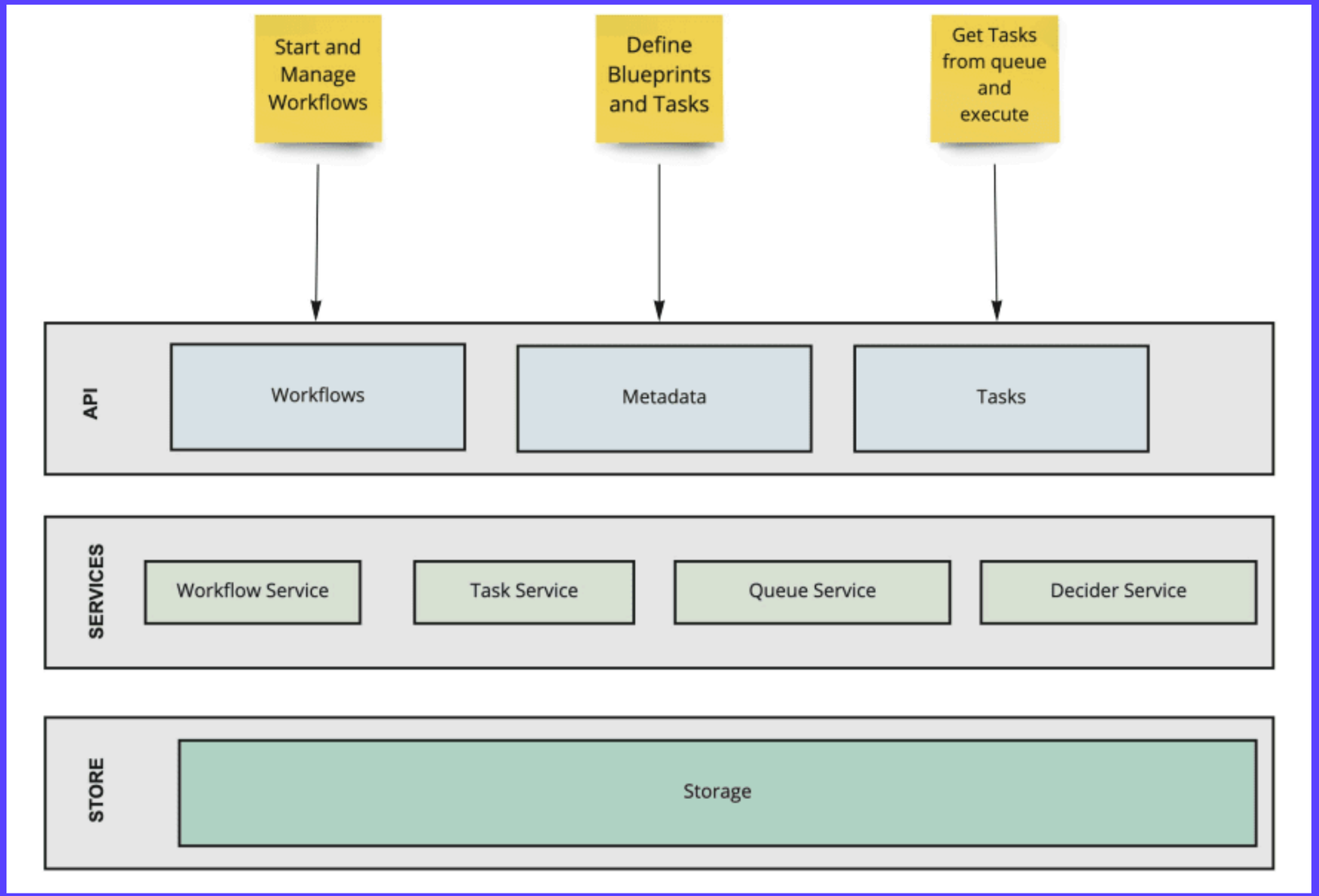
Архитектура



1.- Poll Task
GET /tasks/poll/batch/{*taskType*}

2.- Update Task
POST /tasks

Архитектура



Стек технологий

Persistence

Queue

Payload Storage

ВИДЫ РАСШИРЕНИЙ

Event Provider

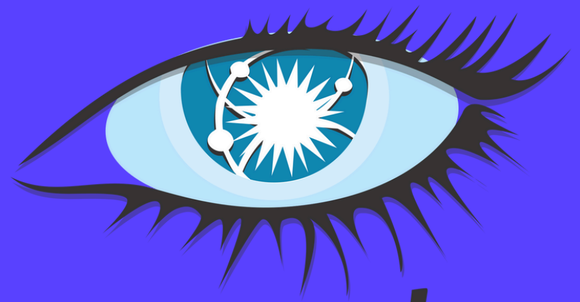
API

Metrics

Persistence

Хранение мета данных и процессов в состоянии IN PROGRESS

Persistence



redis

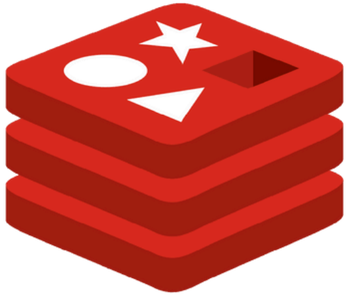
PostgreSQL

MySQL™

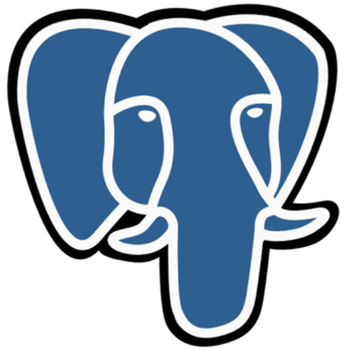
Queue

Хранение и обработка внутренних и внешних задач **Conductor**

Queue



redis



PostgreSQL

Payload Storage

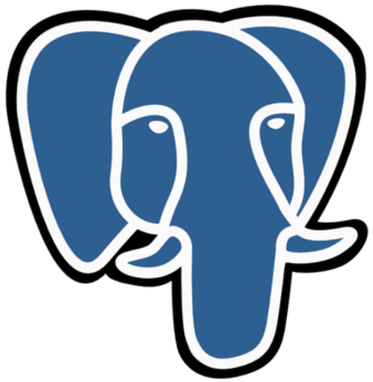
Хранение объектов больше 3-5 MB

Payload Storage

Amazon S3

Azure

BLOB STORAGE



PostgreSQL

Event Provider

Чтение внешних очередей для запуска Event Handler

Event Provider

amazon
SQS

AMQP
Advanced Message Queuing Protocol

NATS

API

Протокол для вызова **Conductor API**

API

Spring Boot
REST API

Metrics

Система экспорта метрик

Metrics

Prometheus

DATADOG

LOG4J

Workflow Definition

Task Definition

Сущности Conductor

Event Handler

Execution

Workflow Definition

Схема рабочего процесса с настройками и списком задач

Workflow Definition


```
{
  "name": "test_workflow",
  "description": "Workflow description",
  "version": 1,
  "tasks": [
    {
      "name": "send_email",
      "taskReferenceName": "send_email",
      "inputParameters": {
        "email": "example@example.com"
      },
      "type": "SIMPLE"
    }
  ]
}
```

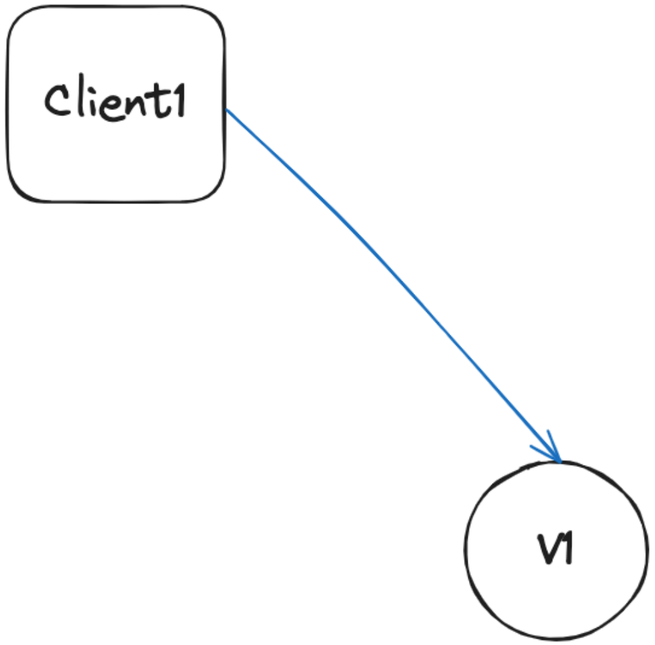
Workflow Definition


```
{
  "name": "test_workflow",
  "description": "Workflow description",
  "version": 1,
  "tasks": [
    {
      "name": "send_email",
      "taskReferenceName": "send_email",
      "inputParameters": {
        "email": "example@example.com"
      },
      "type": "SIMPLE"
    }
  ]
}
```

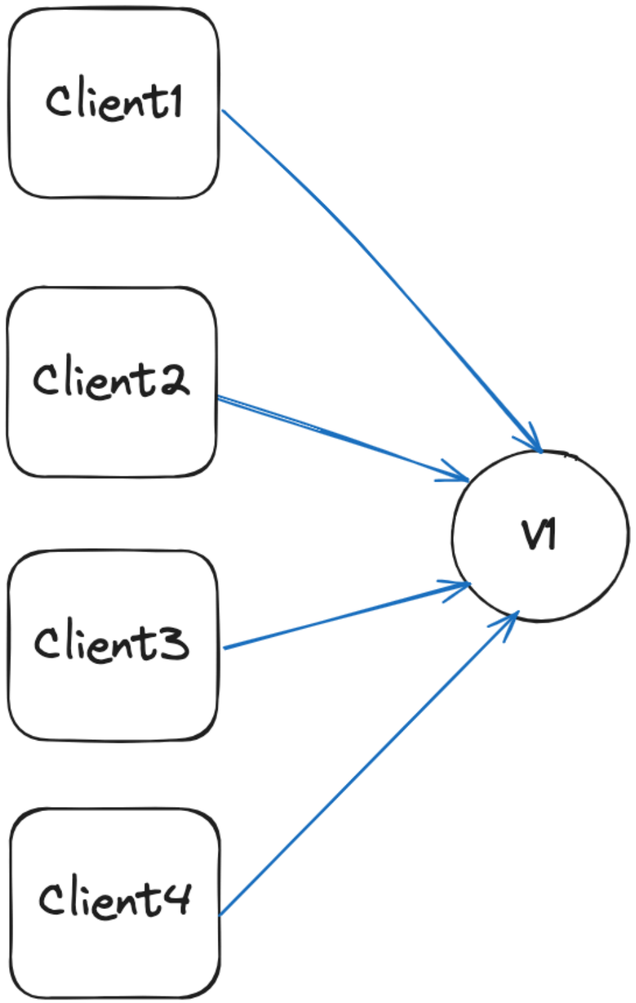
Workflow Definition


```
{
  "name": "test_workflow",
  "description": "Workflow description",
  "version": 1,
  "tasks": [
    {
      "name": "send_email",
      "taskReferenceName": "send_email",
      "inputParameters": {
        "email": "example@example.com"
      },
      "type": "SIMPLE"
    }
  ]
}
```

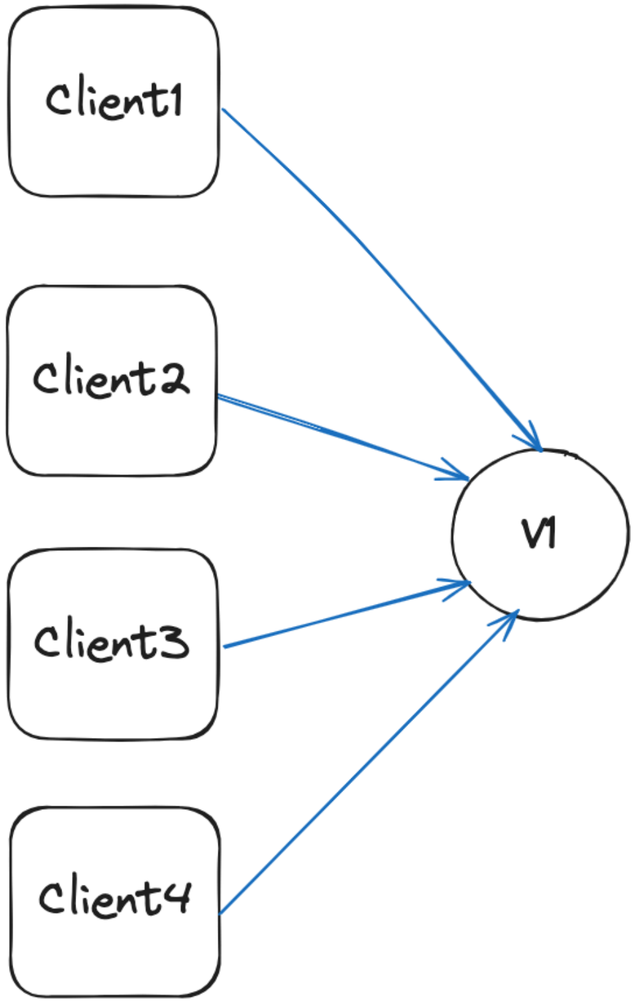
Канареечный деплой

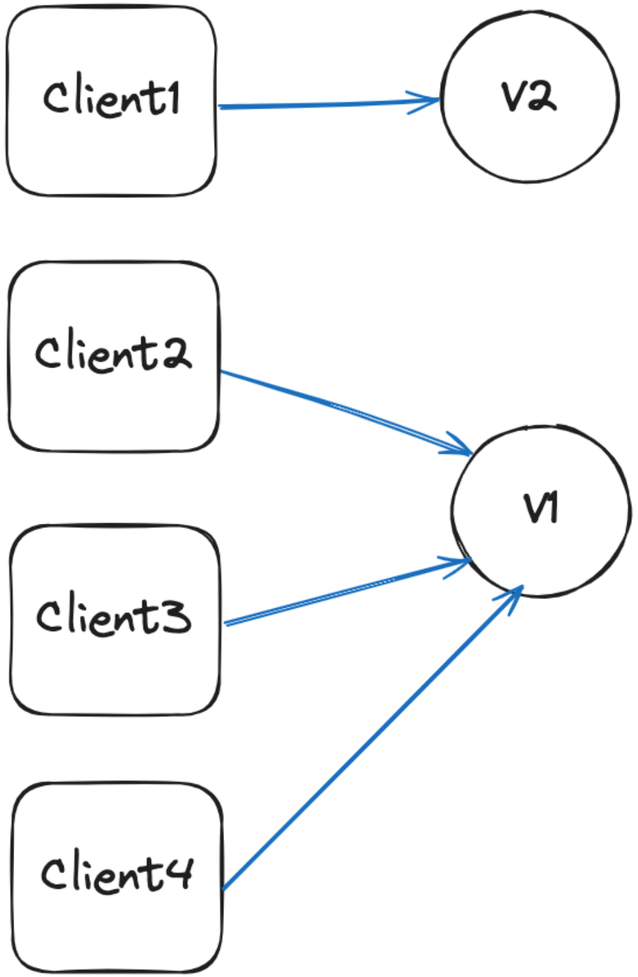


Канареечный деплой



Канареечный деплой





Workflow Definition


```
{
  "name": "test_workflow",
  "description": "Workflow description",
  "version": 1,
  "tasks": [
    {
      "name": "send_email",
      "taskReferenceName": "send_email",
      "inputParameters": {
        "email": "example@example.com"
      },
      "type": "SIMPLE"
    }
  ]
}
```

Workflow Definition


```
{
  "name": "test_workflow",
  "description": "Workflow description",
  "version": 1,
  "tasks": [
    {
      "name": "send_email",
      "taskReferenceName": "send_email",
      "inputParameters": {
        "email": "example@example.com"
      },
      "type": "SIMPLE"
    }
  ]
}
```

Workflow Definition


```
{
  "name": "test_workflow",
  "description": "Workflow description",
  "version": 1,
  "tasks": [
    {
      "name": "send_email",
      "taskReferenceName": "send_email",
      "inputParameters": {
        "email": "example@example.com"
      },
      "type": "SIMPLE"
    }
  ]
}
```

Workflow Definition


```
{
  "name": "test_workflow",
  "description": "Workflow description",
  "version": 1,
  "tasks": [
    {
      "name": "send_email",
      "taskReferenceName": "send_email",
      "inputParameters": {
        "email": "example@example.com"
      },
      "type": "SIMPLE"
    }
  ]
}
```

Workflow Definition


```
[
  {
    "name": "send_email",
    "taskReferenceName": "send_email",
    "inputParameters": {
      "email": "example@example.com"
    },
    "type": "SIMPLE"
  },
  {
    "name": "send_status",
    "taskReferenceName": "send_status_ref",
    "inputParameters": {
      "status": "OK",
      "email_id": "${send_email.output.email_id}"
    },
    "type": "SIMPLE"
  }
]
```

Task Definition

Схема задач, для выполнения внешним worker

Task Definition


```
{
  "name": "send_email",
  "description": "Sending email task def",
  "retryLogic": "FIXED",
  "retryCount": 1,
  "retryDelaySeconds": 5,
  "responseTimeoutSeconds": 20,
  "timeoutSeconds": 60,
  "timeoutPolicy": "TIME_OUT_WF"
}
```

Task Definition


```
{
  "name": "send_email",
  "description": "Sending email task def",
  "retryLogic": "FIXED",
  "retryCount": 1,
  "retryDelaySeconds": 5,
  "responseTimeoutSeconds": 20,
  "timeoutSeconds": 60,
  "timeoutPolicy": "TIME_OUT_WF"
}
```

Task Definition


```
{
  "name": "send_email",
  "description": "Sending email task def",
  "retryLogic": "FIXED",
  "retryCount": 1,
  "retryDelaySeconds": 5,
  "responseTimeoutSeconds": 20,
  "timeoutSeconds": 60,
  "timeoutPolicy": "TIME_OUT_WF"
}
```

Task Definition


```
{
  "name": "send_email",
  "description": "Sending email task def",
  "retryLogic": "FIXED",
  "retryCount": 1,
  "retryDelaySeconds": 5,
  "responseTimeoutSeconds": 20,
  "timeoutSeconds": 60,
  "timeoutPolicy": "TIME_OUT_WF"
}
```

Создание Worker

```
@WorkerTask("send_email")
public @OutputParam("email_id") String sendEmailTask(
        @InputParam("email") String email
) {
    return emailService.sendEmail(email);
}
```

Создание Worker

```
@WorkerTask("send_email")
public @OutputParam("email_id") String sendEmailTask(
        @InputParam("email") String email
) {
    return emailService.sendEmail(email);
}
```

Создание Worker

```
@WorkerTask("send_email")
public @OutputParam("email_id") String sendEmailTask(
        @InputParam("email") String email
) {
    return emailService.sendEmail(email);
}
```

Создание Worker

```
@WorkerTask("send_email")
public @OutputParam("email_id") String sendEmailTask(
        @InputParam("email") String email
) {
    return emailService.sendEmail(email);
}
```

Создание Worker

```
@WorkerTask("send_email")
public @OutputParam("email_id") String sendEmailTask(
        @InputParam("email") String email
) {
    return emailService.sendEmail(email);
}
```

Создание Worker

```python
@worker_task(task_definition_name='get_user_email')
def get_user_email(userid: str) -> str:
    return f'{userid}@example.com'
```

Event Handler

Обработчик внешних событий, запускает **Workflow** или завершает **Task**

Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

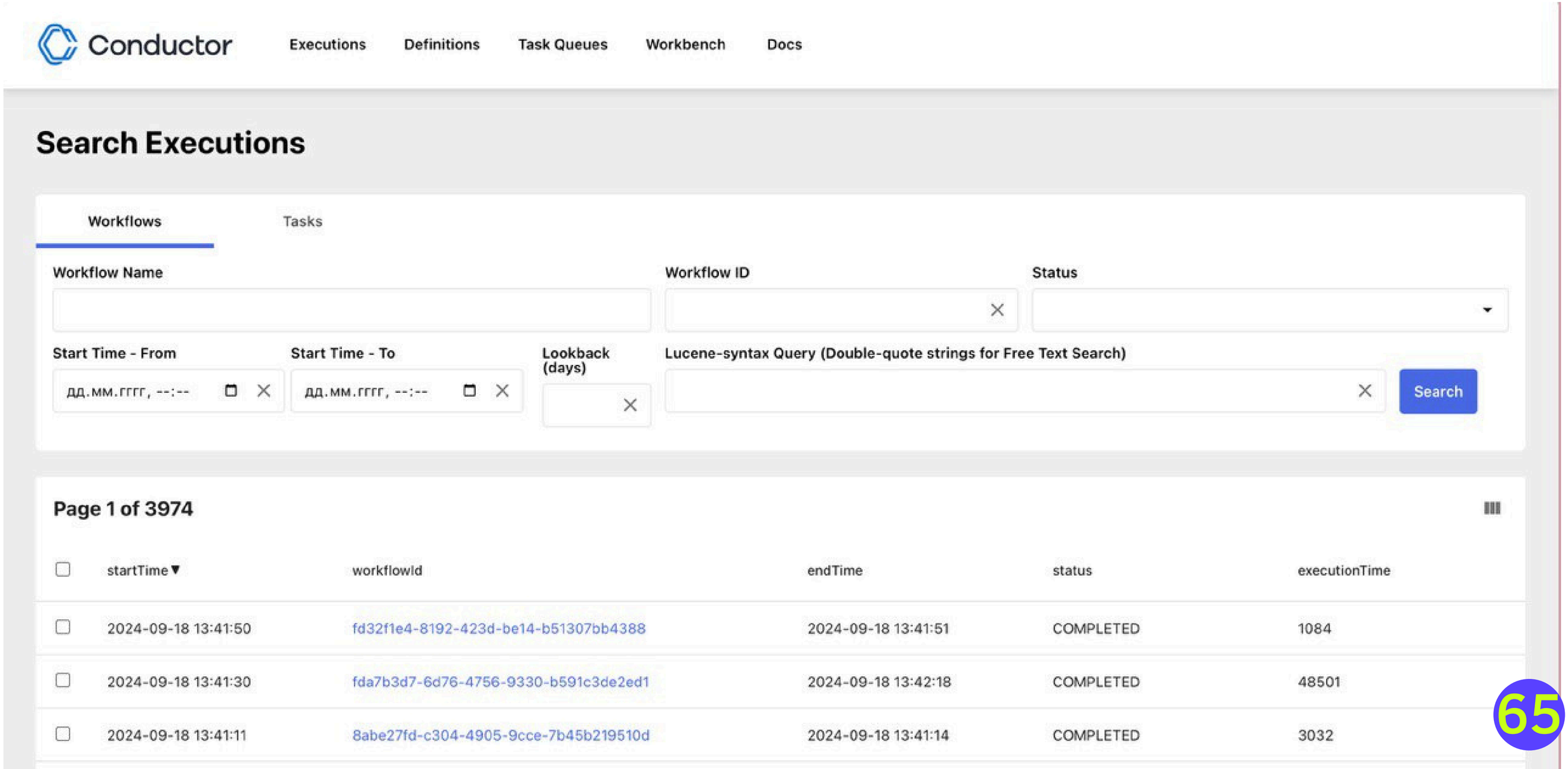
Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

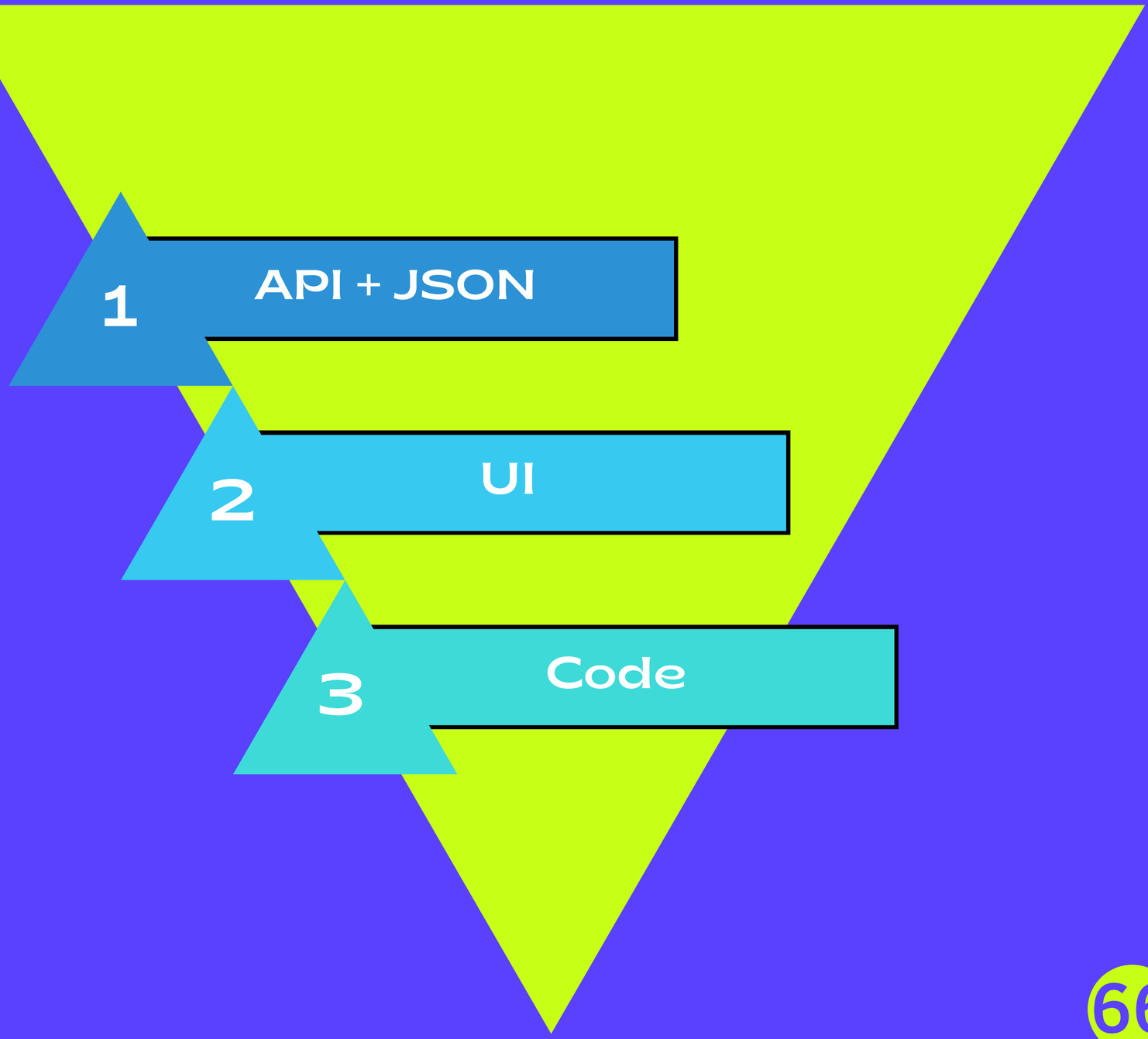
Event Handler


```
{
  "name": "test_complete_task_event",
  "event": "sqs:topicName",
  "actions": [
    {
        "action": "complete_task",
        "complete_task": {
            "workflowId": "${sourceWorkflowId}",
            "taskRefName": "taskNameTest"
        }
    }
  ],
  "active": true
}
```

Conductor UI



СОЗДАНИЕ **WORKFLOW**



Создание Workflow

```
val conductorWorkflow: ConductorWorkflow<Any> =
        WorkflowBuilder<Any>(workflowExecutor)
        .name("sdk_workflow_example")
        .version(1)
        .add(SimpleTask("send_email", "send_email"))
        .build()
```

Создание Workflow

```
val conductorWorkflow: ConductorWorkflow<Any> =
        WorkflowBuilder<Any>(workflowExecutor)
        .name("sdk_workflow_example")
        .version(1)
        .add(SimpleTask("send_email", "send_email"))
        .build()
```

Создание Workflow

```
val conductorWorkflow: ConductorWorkflow<Any> =
        WorkflowBuilder<Any>(workflowExecutor)
        .name("sdk_workflow_example")
        .version(1)
        .add(SimpleTask("send_email", "send_email"))
        .build()
```

Создание Workflow

```
val conductorWorkflow: ConductorWorkflow<Any> =
        WorkflowBuilder<Any>(workflowExecutor)
        .name("sdk_workflow_example")
        .version(1)
        .add(SimpleTask("send_email", "send_email"))
        .build()
```

Создание Workflow

```
val conductorWorkflow: ConductorWorkflow<Any> =
        WorkflowBuilder<Any>(workflowExecutor)
        .name("sdk_workflow_example")
        .version(1)
        .add(SimpleTask("send_email", "send_email"))
        .build()
```

Создание Workflow

```
def create_workflow():
    wf = ConductorWorkflow(name='test_workflow', version=1,
                           executor=workflow_executor)
    send_email = SimpleTask(task_ref_name='send_email',http_input={
        'email': 'example@example.com'
    })
    wf >> send_email
```

Создание Workflow

```
def create_workflow():
    wf = ConductorWorkflow(name='test_workflow', version=1,
                           executor=workflow_executor)
    send_email = SimpleTask(task_ref_name='send_email',http_input={
        'email': 'example@example.com'
    })
    wf >> send_email
```

Создание Workflow

```
def create_workflow():
    wf = ConductorWorkflow(name='test_workflow', version=1,
                           executor=workflow_executor)
    send_email = SimpleTask(task_ref_name='send_email',http_input={
        'email': 'example@example.com'
    })
    wf >> send_email
```

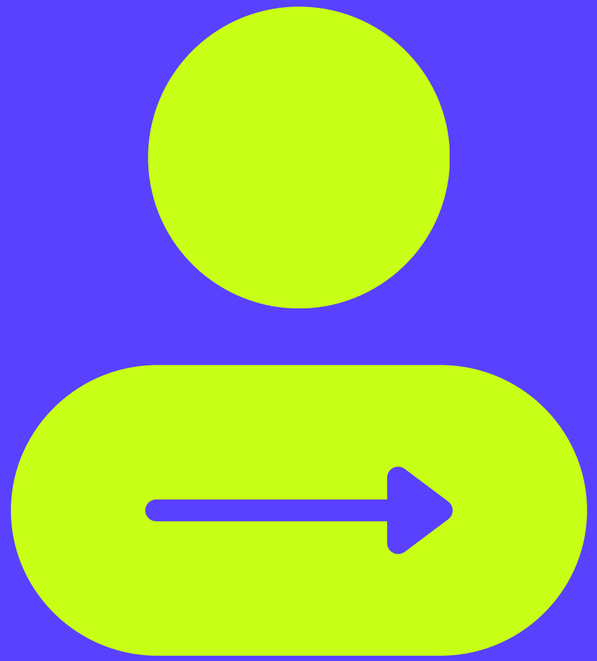
Создание Workflow

```
def create_workflow():
    wf = ConductorWorkflow(name='test_workflow', version=1,
                           executor=workflow_executor)
    send_email = SimpleTask(task_ref_name='send_email',http_input={
        'email': 'example@example.com'
    })
    wf >> send_email
```

Создание Workflow

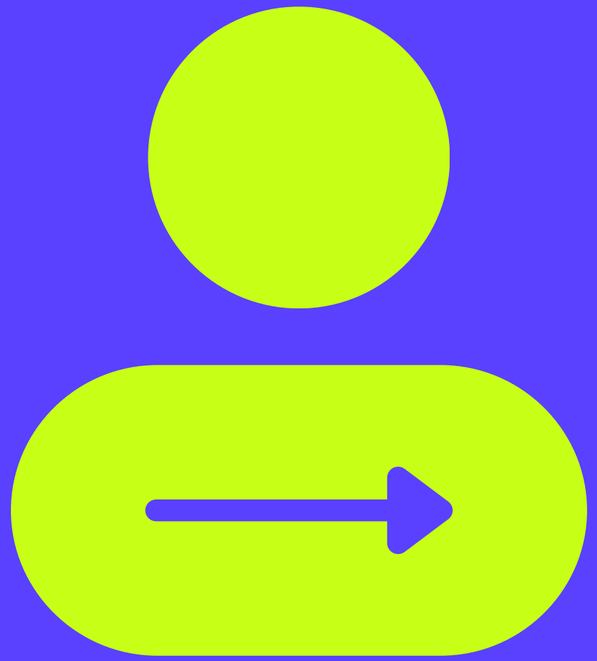
```
def create_workflow():
    wf = ConductorWorkflow(name='test_workflow', version=1,
                           executor=workflow_executor)
    send_email = SimpleTask(task_ref_name='send_email',http_input={
        'email': 'example@example.com'
    })
    wf >> send_email
```

Чего не хватило нам?



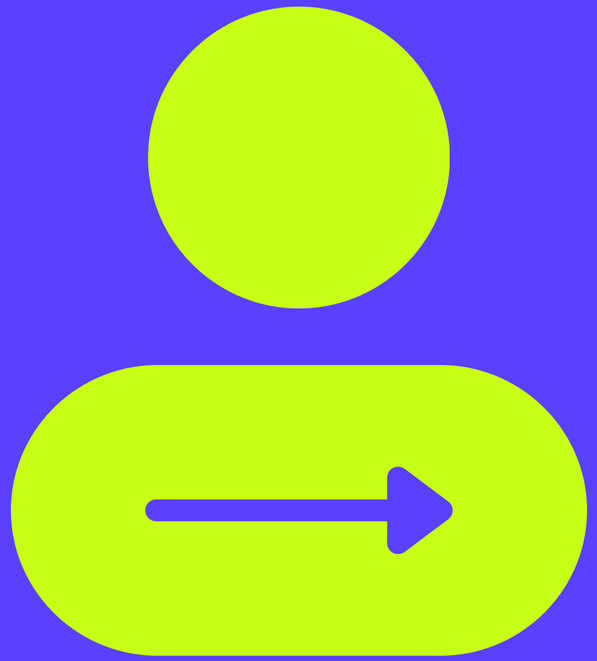
RBAC + Masking

Чего не хватило нам?



Workflow
Ratelimit

Чего не хватило нам?



Sync API

Добавление модуля

```
@Configuration
@ConditionalOnProperty("conductor.security.bearer.enabled",
    matchIfMissing = false)
open class SecurityBearerConfiguration {

    @Bean
    open fun filterChain(http: HttpSecurity): SecurityFilterChain {
        ...
    }
}
```

Добавление модуля

```
@Configuration
@ConditionalOnProperty("conductor.security.bearer.enabled",
    matchIfMissing = false)
open class SecurityBearerConfiguration {

    @Bean
    open fun filterChain(http: HttpSecurity): SecurityFilterChain {
        ...
    }
}
```

Добавление модуля

```
@Configuration
@ConditionalOnProperty("conductor.security.bearer.enabled",
    matchIfMissing = false)
open class SecurityBearerConfiguration {

    @Bean
    open fun filterChain(http: HttpSecurity): SecurityFilterChain {
        ...
    }
}
```

Добавление модуля

```
@Configuration
@ConditionalOnProperty("conductor.security.bearer.enabled",
    matchIfMissing = false)
open class SecurityBearerConfiguration {

    @Bean
    open fun filterChain(http: HttpSecurity): SecurityFilterChain {
        ...
    }
}
```

Пример настройки oauth2

```
@Bean
open fun filterChain(http: HttpSecurity): SecurityFilterChain {
    http.run {
        authorizeHttpRequests {
            it.requestMatchers("/api/admin/**").hasRole("ADMIN")
        }
        authorizeHttpRequests { it.anyRequest().permitAll() }

        oauth2ResourceServer { config ->
            config.jwt { it.jwkSetUri("https://idp.example.com/.well-known/jwks.json") }
        }
        csrf { it.disable() }
    }
    return http.build()
}
```

Пример WorkflowListener

```
public class JournalListener implements WorkflowStatusListener {

    @Override
    public void onWorkflowCompleted(WorkflowModel workflowModel) {
        if (isJournalEnabled(workflowModel))
            journalService.sendJournal(
                    workflowToMessage(workflowModel));
    }
}
```

Пример WorkflowListener

```
public class JournalListener implements WorkflowStatusListener {

    @Override
    public void onWorkflowCompleted(WorkflowModel workflowModel) {
        if (isJournalEnabled(workflowModel))
            journalService.sendJournal(
                    workflowToMessage(workflowModel));
    }
}
```

Пример WorkflowListener

```
public class JournalListener implements WorkflowStatusListener {

    @Override
    public void onWorkflowCompleted(WorkflowModel workflowModel) {
        if (isJournalEnabled(workflowModel))
            journalService.sendJournal(
                    workflowToMessage(workflowModel));
    }
}
```

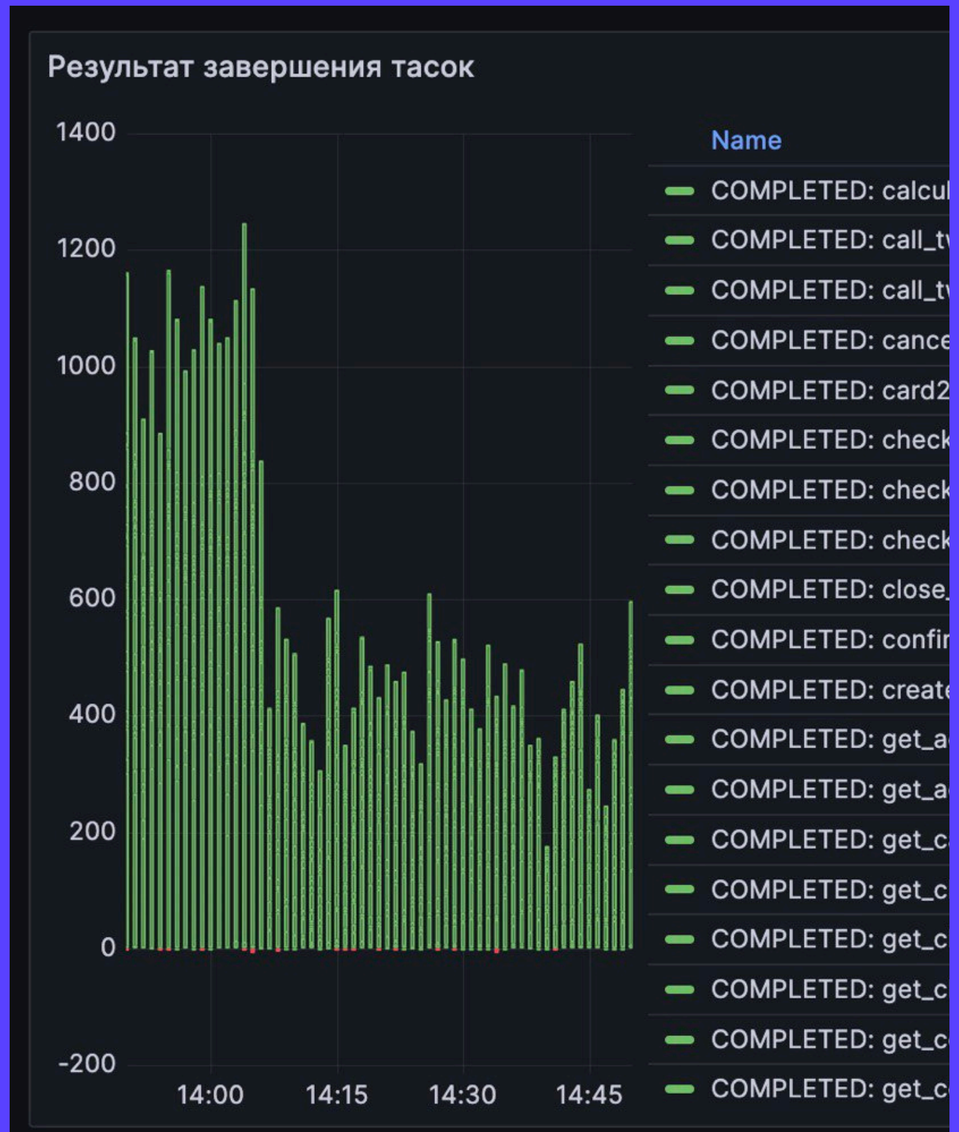
Пример WorkflowListener

```
public class JournalListener implements WorkflowStatusListener {

    @Override
    public void onWorkflowCompleted(WorkflowModel workflowModel) {
        if (isJournalEnabled(workflowModel))
            journalService.sendJournal(
                    workflowToMessage(workflowModel));
    }
}
```

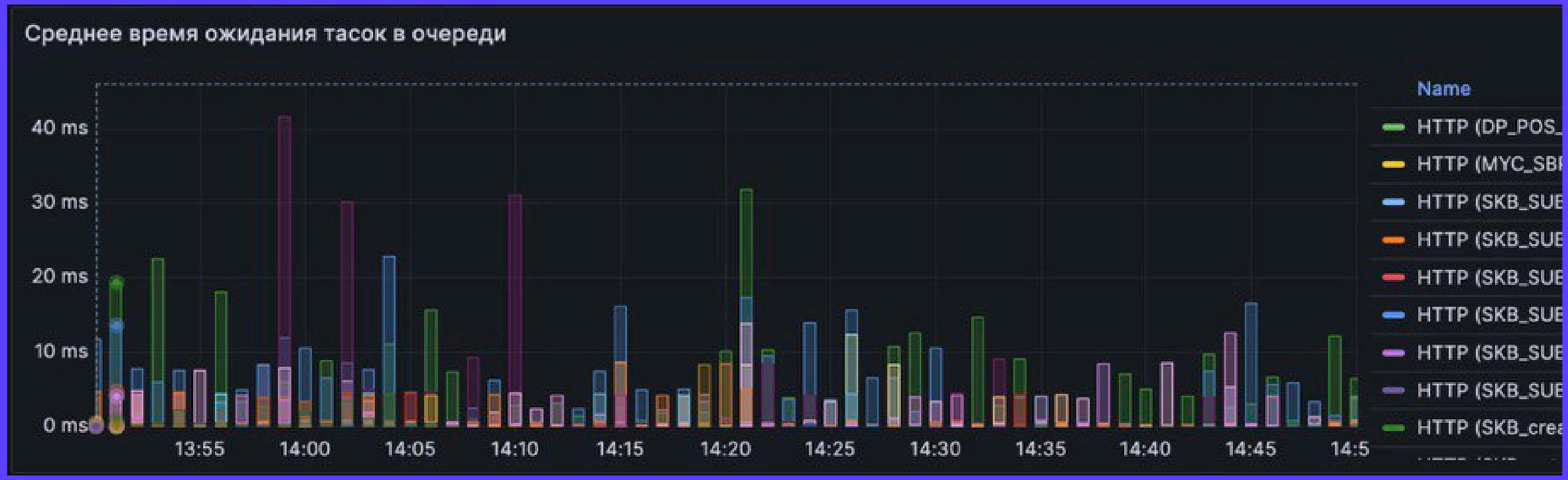
Пример WorkflowListener

```
public class JournalListener implements WorkflowStatusListener {

    @Override
    public void onWorkflowCompleted(WorkflowModel workflowModel) {
        if (isJournalEnabled(workflowModel))
            journalService.sendJournal(
                    workflowToMessage(workflowModel));
    }
}
```

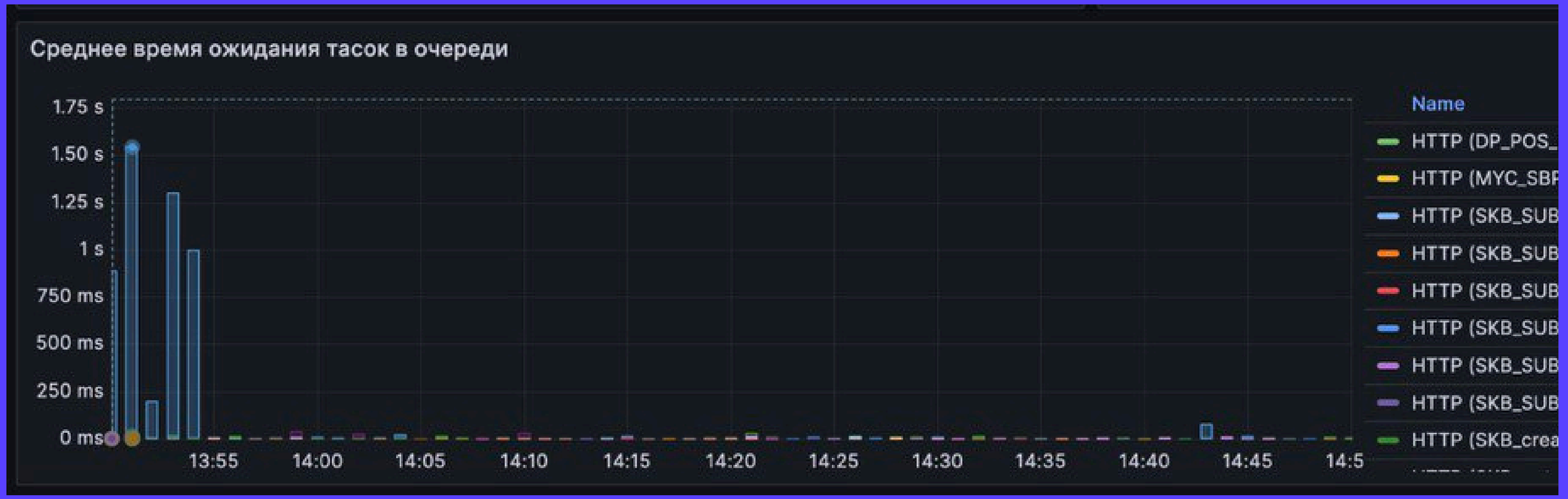
Мониторинг


Результат завершения тасок
1400
1200
1000
800
600
400
200
0
-200
14:00
14:15
14:30
14:45
Name
COMPLETED: calcul
COMPLETED: call_t
COMPLETED: call_t
COMPLETED: cance
COMPLETED: card2
COMPLETED: check
COMPLETED: check
COMPLETED: check
COMPLETED: close_
COMPLETED: confir
COMPLETED: create
COMPLETED: get_a
COMPLETED: get_a
COMPLETED: get_c
COMPLETED: get_c
COMPLETED: get_c
COMPLETED: get_c
COMPLETED: get_c
COMPLETED: get_c

Мониторинг

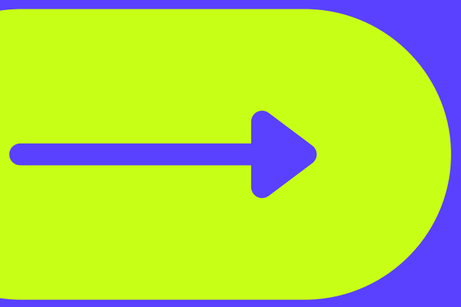

Среднее время ожидания тасок в очереди
40 ms
30 ms
20 ms
10 ms
0 ms
13:55
14:00
14:05
14:10
14:15
14:20
14:25
14:30
14:35
14:40
14:45
14:5
Name
HTTP (DP_POS_
HTTP (MYC_SBI
HTTP (SKB_SUE
HTTP (SKB_SUE
HTTP (SKB_SUE
HTTP (SKB_SUE
HTTP (SKB_SUE
HTTP (SKB_SUE
HTTP (SKB_crea

Мониторинг


Среднее время ожидания тасок в очереди
1.75 s
1.50 s
1.25 s
1 s
750 ms
500 ms
250 ms
0 ms
13:55
14:00
14:05
14:10
14:15
14:20
14:25
14:30
14:35
14:40
14:45
14:5
Name
HTTP (DP_POS_
HTTP (MYC_SBP
HTTP (SKB_SUB
HTTP (SKB_SUB
HTTP (SKB_SUB
HTTP (SKB_SUB
HTTP (SKB_SUB
HTTP (SKB_SUB
HTTP (SKB_crea

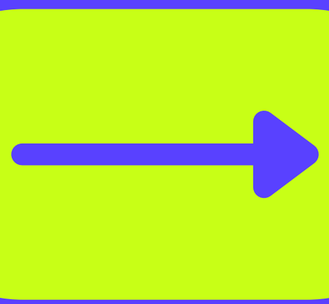
Нюансы

Маппинг данных

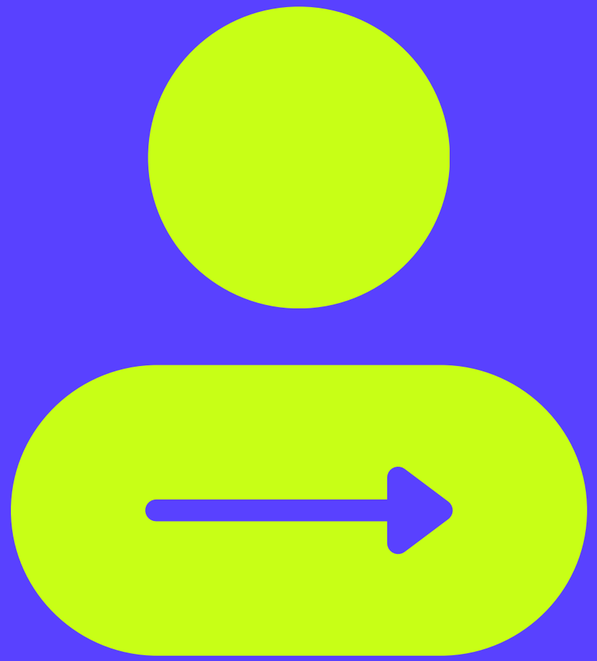


Нюансы

Много настроек

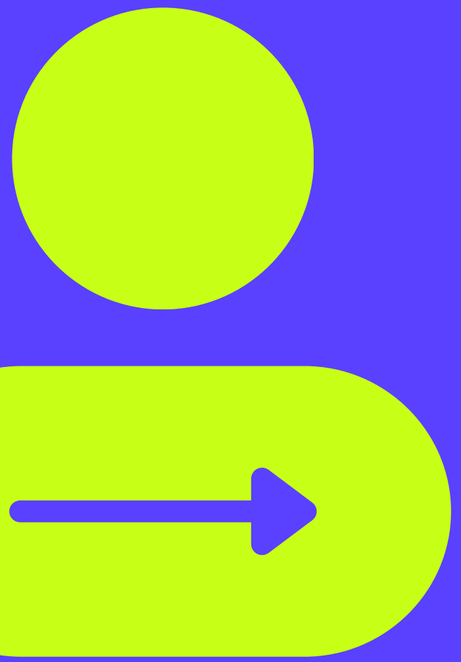


Нюансы



Время выполнения Workflow

Нюансы



SDK + Security

ИТОГО

\+

\–

ПЛЮСЫ

Удобный и красивый интерфейс

Реализован на **Spring + Java**

Легко добавить функционал

Низкий порог входа

МИНУСЫ

Маппинг данных

Время выполнения **Workflow**

Авторизация в **SDK**

Подбор множества настроек

Conductor

Задать вопрос

Попробуйте сами!