

Paslaugos užsakymo eigos API

Aprašymas

Konceptualios architektūros elementų atitikmuo

Swagger dokumentacija

Implementacija

Vidinės bibliotekos

NestJS moduliai

Komunikacija su mikroservisais

RabbitMQ

AWS S3

Autentifikacija

Environment variables

Operacijų žurnalas (Logging)

OpenTelemetry

Request Context

Docker konteineris

Bazinė informacija

Health check

Aprašymas

Viešoji REST API skirta e-paslaugų užsakymų pildymui ir valdymui. API veikia kaip **gateway** išorinėms sistemoms (portalams) integracijai su SPP platforma per RabbitMQ.



Konceptualios architektūros elementų atitikmuo

- Veikia kaip gateway perduodant loginį įvykdymą į mikroservisų sluoksnį šiem API:
 - Paslaugų užsakymo eigos API
 - Naudotojo užsakymų API
 - Naudotojo užsakymų rezultatų API

Swagger dokumentacija

Pilna API endpoint'ų dokumentacija prieinama per Swagger UI:

URL: <http://workflows-api.demo.spp.codigi.lt>

Swagger URL: <http://workflows-api.demo.spp.codigi.lt/api>

OpenAPI JSON: <http://workflows-api.demo.spp.codigi.lt/api-json>

Implementacija

Visas kodas talpinamas `/apps/e-service-workflows-api` aplanke.

Įgyvendinta naudojantis **NestJS** framework (Node.js).

Vidinės bibliotekos

Biblioteka	Paskirtis
<code>@spp/types</code>	Bendri TypeScript tipai ir DTO
<code>@spp/ms-clients</code>	RabbitMQ klientų moduliai komunikacijai su MS
<code>@spp/ms-utils</code>	Bendros utilitos (logging, file storage)
<code>@spp/utils</code>	Bendri utilities

NestJS moduliai

Modulis	Kelias	Aprašymas
<code>EServicesModule</code>	<code>v1.0/e-services</code>	E-paslaugų duomenų skaitymas
<code>EServiceWorkflowsModule</code>	-	Workflow operacijų vykdymas
<code>EServiceRequestsModule</code>	-	E-paslaugų užsakymų valdymas
<code>EServiceRequestsProfilesModule</code>	<code>v1.0/e-service-requests-profiles/*</code>	Naudotojų ir gavėjų profiliai

FilesModule	-	Failų įkėlimas ir atsissiuntimas
StandardizedListsModule	v1.0/standardized-lists	Standartizuotų sąrašų prieiga
EventsModule	v1.0/events	Įvykių sekimas
InstitutionsModule	-	Institucijų duomenų skaitymas
InstitutionsEServicesModule	-	E-paslaugos su institucijų info
NotificationsModule	-	Pranešimų gavimas
TasksModule	-	Paslaugų gavėjo / institucijos darbuotojo / asmens užduotys
AuthModule	-	Autentifikacijos guard
HealthModule	healthz	Health check endpoints

Komunikacija su mikroservisais

RabbitMQ

API komunikuoja su mikroservisais per **RabbitMQ** naudojant `@spp/ms-clients` biblioteką.

Queue naming pattern: `{ENV_PREF}_{queue_name}`

Kur `ENV_PREF` yra aplinkos prefiksas (pvz., `loc`, `dev`, `sta`, `prod`).

MS Client modulis	RabbitMQ Queue	Mikroservisas
EServicesClientModule	<code>{ENV_PREF}_e_services_queue</code>	<code>e-services</code>
EServiceWorkflowsClientModule	<code>{ENV_PREF}_e_service_workflows_queue</code>	<code>e-service-workflows</code>

EServiceRequestsClientModule	{ENV_PREF}_e_service_requests_queue	e-service-requests
EServiceRequestsProfilesClientModule	{ENV_PREF}_e_service_request_profiles_queue	e-service-requests-profiles
FilesClientModule	{ENV_PREF}_files_queue	files
StandardizedListsClientModule	{ENV_PREF}_standardized-lists	standardized-lists
EventsClientModule	{ENV_PREF}_events_queue	events
InstitutionsClientModule	{ENV_PREF}_institutions_queue	institutions
NotificationsClientModule	{ENV_PREF}_notifications_queue	notifications

AWS S3

Failų saugojimui naudojama **AWS S3** per `FileStorageModule` iš `@spp/ms-utils`.

Autentifikacija

Dauguma endpoint'ų **nereikalauja autentifikacijos** - saugumas bus valdomas Gravitee apribojimais tarp SPK ir SPP API.

Guard	Paskirtis
AuthGuard	JWT Bearer token validacija (naudojamas pasirinktinai simulatoriuje)

Environment variables

Kintamasis	Aprašymas	Pavyzdys
------------	-----------	----------

ENV_PREF	Aplinkos prefiksas	local , dev , sta , prod
NODE_ENV	Node.js aplinka	development , production
PORT	API portas	3000
RMQ_CONNECTION_STRING	RabbitMQ prisijungimo eilutė	amqp://user:password@spp-mq-rabbitmq:5672
REDIS_HOST	Redis serverio adresas	spp-cache-redis
REDIS_PORT	Redis portas	6379
DB_HOST	PostgreSQL serverio adresas	spp-db-postgres
DB_PORT	PostgreSQL portas	5432
DB_USERNAME	DB vartotojo vardas	root
DB_PASSWORD	DB slaptažodis	password
DB_DB	Duomenų bazės pavadinimas	spp
AWS_S3_REGION	AWS S3 regionas	eu-north-1
AWS_S3_BUCKET_NAME	S3 bucket pavadinimas	spp-loc-ms-files
AWS_S3_ACCESS_KEY_ID	AWS access key ID	(secret)
AWS_S3_SECRET_ACCESS_KEY	AWS secret access key	(secret)
GRAYLOG_HOST	Graylog serverio adresas	graylog
GRAYLOG_PORT	Graylog portas	12201
GRAYLOG_PROTOCOL	Graylog protokolas	udp

OTEL_EXPORTER_OTLP_ENDPOINT	OpenTelemetry collector endpoint	otel-collector:4317
OTEL_EXPORTER_OTLP_PROTOCOL	OpenTelemetry protokolas	grpc
OTEL_SERVICE_NAME	Serviso pavadinimas telemetrijai	e-service-workflows-api
OTEL_SERVICE_VERSION	Serviso versija telemetrijai	1.0.0
OTEL_TRACES_EXPORTER	Traces eksporteris	otlp
OTEL_LOGS_EXPORTER	Logs eksporteris	otlp

Operacijų žurnalas (Logging)

OpenTelemetry

Naudojamas **OpenTelemetry** centralizuotam logų ir traces rinkimui.

Implementacija:

- `HttpRequestLoggingInterceptor` - HTTP užklausų/atsakymų logavimas
- `LoggerModule` iš `@spp/ms-utils` - centralizuotas logging
- Logs siunčiami į OpenTelemetry Collector per OTLP HTTP protokolą (`/v1/logs`)
- Collector persiunčia logs į **Graylog**

Loguojami duomenys:

- HTTP method, URL, status code
- Request/response duration (ms)
- Request ID (per `nestjs-cls`)
- Error details (message, name, stack)
- User agent, client IP

Request Context

`nestjs-cls` naudojamas request context tracking:

- Automatinis `X-Request-Id` propagavimas

Docker konteineris

Bazinė informacija

Parametras	Reikšmė
Darbo direktorija	<code>/workspace/apps/e-service-workflows-api</code>
Portas	<code>3000</code> (mapped to <code>3058</code> in docker-compose)
Paleidimo komanda	<code>pnpm run start:dev</code>

Health check

```
1 | curl -fsSL http://localhost:3058/healthz
```