Contents

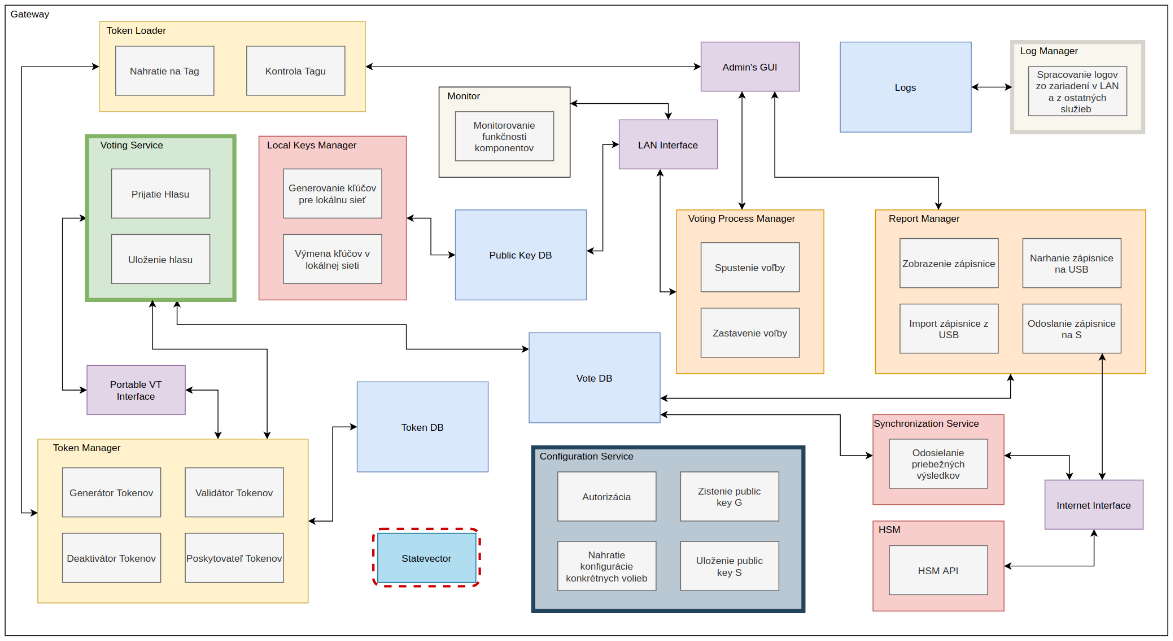
1. Gateway

Gataway je zariadenie nachádzajúce sa vo velebnej miestnosti. V miestnosti sa nachádza vždy len jeden gateway. Zabezpečuje komunikáciu medzi volebnými terminálmi a serverom. Gateway obsahuje lokálnu databázu pre hlasy aj tokeny, takže dokáže fungovať aj bez pripojenia k internetu a vie urobiť synchronizáciu na inom mieste, kde je internet dostupný.

Gataway sa má nachádzať na chránenom mieste a pristupovať k nemu smú iba členovia volebnej komisie napríklad pri spustení alebo zastavení volieb alebo nahrávaní tokenov na NFC tagy.

1.1 Architektúra

popis architektury



1.2 Mikroslužby a ich smerovanie

V nasledujúcej tabuľke uvádzame zoznam mikroslužieb a statických súborv na gateway-i a ich smerovanie.

Service	Path
Voting service	`/voting-service-api/`
Synchronization service	`/synchronization-service-api/`
Voting process manager	`/voting-process-manager-api/`
Token manager	`/token-manager-api/`
State vector	`/statevector/`
config.json	`/statevector/config/config.json`
datamodels.yaml	`/statevector/config/datamodels.yaml`

1.3 State vector

Služba zodpovedná za udržiavanie aktuálneho stavu gateway-u.

Udržuje tieto stavy: - `state_election` - stav volieb - `state_write` - stav zapisovačky - `state_register_terminals` - stav registrácie terminálov - `office_id` - id volebnej miestnosti - `pin` - pin kód k GUI aplikácii na gataway-i - `server_key` - verejný kľúč servera - `server_address` - adresa servera

1.3.1 Konfiguračný súbor

Konfiguračný súbor obsahuje celú konfiguráciu volieb pre konkrétnu volebnú miestnosť. Je dostupný ako statický súbor na adrese `/statevector/config/config.json` pomocou Nginx.

1.3.2 Popis API

hello___get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/', headers = headers)

print(r.json())
```

`GET /`

Hello

Sample testing endpoint

Example responses

200 Response

```
{
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

get__state__election__state__election__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/state_election', headers =
    headers)

print(r.json())
```

`GET /state_election`

Get State Election

Get election state string 0 or 1

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

set__state__election__state__election__post

Code samples

```python
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/statevector/state_election', headers =
   headers)

print(r.json())
```

`POST /state_election`

Set State Election

Set election state string 0 or 1

Body parameter

```
"string"
```

Parameters

Name	In	Type	Required	Description
body	body	string	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

get__state_write_state_write__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/state_write', headers =
   headers)

print(r.json())
```

`GET /state_write`

Get State Write

Get write state string 0 or 1

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

set__state_write_state_write__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}
```

```
r = requests.post('/gateway/statevector/state_write', headers =
    headers)

print(r.json())
```

`POST /state_write`

Set State Write

Set write state string 0 or 1

Body parameter

```
"string"
```

Parameters

Name	In	Type	Required	Description
body	body	string	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

state__register__terminals__state__register__terminals__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/state_register_terminals',
    headers = headers)

print(r.json())
```

`GET /state_register_terminals`

State Register Terminals

Get terminals registration state string 0 or 1

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

set__state__register__terminals__state__register__terminals__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/statevector/state_register_terminals',
   headers = headers)

print(r.json())
```

`POST /state_register_terminals`

Set State Register Terminals

Set register terminals state string 0 or 1

Body parameter

```
"string"
```

Parameters

Name	In	Type	Required	Description
body	body	string	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

get__office__id__office__id__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/office_id', headers = headers)

print(r.json())
```

`GET /office_id`

Get Office Id

Get office id

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

get__pin__pin__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/pin', headers = headers)

print(r.json())
```

`GET /pin`

Get Pin

Get pin

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

get__server__key__server__key__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/server_key', headers = headers)

print(r.json())
```

`GET /server_key`

Get Server Key

Get server key

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

get__server__address__server__address__get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/statevector/server_address', headers =
    headers)

print(r.json())
```

`GET /server_address`

Get Server Address

Get server address

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

Schemas

1.3.2.12.1 HTTPValidationError

```json
{
  "detail": [
    {
      "loc": [
        "string"
      ],
      "msg": "string",
      "type": "string"
    }
  ]
}
```

HTTPValidationError

Properties

Name	Type	Required	Restrictions	Description
detail	[ValidationError]	false	none	none

1.3.2.12.2 ValidationError

```
{
  "loc": [
    "string"
  ],
  "msg": "string",
  "type": "string"
}
```

ValidationError

Properties

Name	Type	Required	Restrictions	Description
loc	[string]	true	none	none
msg	string	true	none	none
type	string	true	none	none

1.4 Synchronization service

Služba je zodpovedná za sychronizáciu hlasov medzi gateway-om a serverom. Služba je implementovaná ako REST API v knižnici Fast API.

Služba pracuje z hlasmi v lokálnej Mongo databáze, ktoré boli vložené pomocou Voting service. Hlasy sa synchronizujú po dávkach (prednastavená hodnota je 10) a po zaširovaní sa posielajú pomocou HTTP požiadavky na endpoint servera, ktorý ich zvaliduje. Synchronizácia prebehne úspešne iba ak sú všetky hlasy v poriadku prijaté. Úspešne synchronizované hlasy označí ako `{"synchronized": true}`.

Synchronizácia prebieha na pozadí v intervale každú minútu (implementované pomocou Fast API Utils Repeated Tasks). Dá sa však spustiť aj manuálne pomocou endpointu `POST /api/synchronize`, ktorý je popísaný nižšie.

1.4.1 Štruktúra posielaných hlasov

Hlasy sú posielané v HTTP požiadavke, ktorú tvorí json s id volebnej miestnosti a hlasmi, ktoré sú zašifrované ako pole zašifrovaných hlasov pomocou knižnice *rsaelectie*, funkcie `encrypt_vote`.

```
{
  "polling_place_id": 0,
  "votes": [
    {
      "encrypted_message": "string",
      "encrypted_object": "string"
    }
  ]
}
```

1.4.2 Popis API

root___get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/synchronization-service-api/', headers =
    headers)

print(r.json())
```

`GET /`

Root

Simple hello message.

Example responses

200 Response

```json
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

synchronize_synchronize_post

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/gateway/synchronization-service-api/synchronize',
    headers = headers)

print(r.json())
```

`POST /synchronize`

Synchronize

Try to send local votes to server and updates local status. If server response is different than 200, response has status 500 with error from server.

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

statistics__statistics__post

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/gateway/synchronization-service-api/statistics',
    headers = headers)

print(r.json())
```

`POST /statistics`

Statistics

Provide statistics of votes in gateway database. Count of synchronized and unsynchronized votes.

Example responses

200 Response

```
{
  "status": "string",
  "last_synchronization": null,
  "last_success_synchronization": null,
  "statistics": {
    "all_count": 0,
    "syncronized_count": 0,
    "unsyncronized_count": 0
  }
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

seed__seed__post

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/gateway/synchronization-service-api/seed',
    headers = headers)

print(r.json())
```

`POST /seed`

Seed

Insert 10 unsynced dummy votes into gataway local gatabase.

Example responses

200 Response

```
{
  "status": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

test__encrypt__test__encrypt__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}
```

```
r = requests.get('/gateway/synchronization-service-api/test-encrypt',
    headers = headers)

print(r.json())
```

`GET /test-encrypt`

Test Encrypt

Get a batch of encrypted votes.

Example responses

200 Response

```
{
  "polling_place_id": 0,
  "votes": []
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

Token manager

Služba je zodpovedná za generovanie, overovanie a deaktivovanie tokenov nahrávaných na NFC tagy. Služba rovnako ovláda a interaguje s Token writter-om , ktorý sa stará o samostné nahranie tokenu na NFC tag.

Token je generovaný pomocou `uuid` bez znakov `-`, napríklad `858c0eb798a8475dbcf67e29ddb4966e`.

Deaktivovaný token je označený ako `{"active": false}`.

Aktivovaný a zapísaný token je označený ako `{"active": true}` a `{"written": true}`.

Token je považovaný ako platný iba ak je aktívny (`{"active": true}`).

1.4.3 Komunikácia s frontendom

Token manager komunikuje s frontendovou aplikáciou pomocou websocketov. Používateľa informuje o stave zapisovačky o úspešnom alebo neúspešnom zapísaní tokenu alebo o možnosti nahrávania ďalšieho tokenu. Vo websockete sa posiela udalosť `writer_status`, ktorý nadobúda hodnoty `off`, `idle`, `success`, `error`.

1.4.4 Popis API

root___get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/token-manager-api/', headers = headers)

print(r.json())
```

`GET /`

Root

Simple hello message.

Example responses

200 Response

```json
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

activate__state__tokens__writer__activate__post

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r =
   requests.post('/gateway/token-manager-api/tokens/writer/activate',
   headers = headers)

print(r.json())
```

`POST /tokens/writer/activate`

Activate State

Activate NFC writer machine. After turning on, machine's LED will turn on and be able to write data to NFC tokens.

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

deactivate__state__tokens__writer__deactivate__post

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r =
    requests.post('/gateway/token-manager-api/tokens/writer/deactivate',
    headers = headers)

print(r.json())
```

`POST /tokens/writer/deactivate`

Deactivate State

Deactivate NFC writer machine. Led on machine will turn off.

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

delete__unwritten__tokens__writer__delete__post

Code samples

```python
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/token-manager-api/tokens/writer/delete',
   headers = headers)

print(r.json())
```

`POST /tokens/writer/delete`

Delete Unwritten

Delete unwritten NFC tokens from database.

Body parameter

```
{
  "event": "string"
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body__delete__unwritten__tokens__writer__delete__post	true	none

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

update__written__tokens__writer__update__post

Code samples

```python
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/token-manager-api/tokens/writer/update',
   headers = headers)

print(r.json())
```

`POST /tokens/writer/update`

Update Written

Update NFC token state from unwritten to written.

Body parameter

```json
{
  "token": "string"
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body_update_written_tokens_writer_update_post	true	none

Example responses

200 Response

```json
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

create_token_tokens_create_post

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/gateway/token-manager-api/tokens/create', headers
    = headers)

print(r.json())
```

`POST /tokens/create`

Create Token

Generates new token and returns it.

Example responses

200 Response

```
{
  "status": "string",
  "token": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

validate__token__tokens__validate__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/token-manager-api/tokens/validate',
    headers = headers)

print(r.json())
```

`POST /tokens/validate`

Validate Token

Validate if provided token is valid. If token is invalid returns empty response with status 403 else status 200.

Body parameter

```
{
  "token": "string"
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body__validate_token__tokens_validate__post	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

deactivate__token__tokens__deactivate__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/token-manager-api/tokens/deactivate',
    headers = headers)

print(r.json())
```

`POST /tokens/deactivate`

Deactivate Token

Deactivate provided token. Change active status to false. If token is invalid returns empty response with status 403 else status 200.

Body parameter

```
{
  "token": "string"
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body_deactivate_token_tokens_deactivate_post	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

delete_token_tokens_delete_delete

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.delete('/gateway/token-manager-api/tokens/delete',
    headers = headers)

print(r.json())
```

`DELETE /tokens/delete`

Delete Token

Delete provided token. If token is invalid returns empty response with status 403 else status 200.

Body parameter

```
{
  "token": "string"
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body_delete_token_tokens_delete_delete	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

Schemas

1.4.4.10.1 Body__deactivate__token__tokens__deactivate__post

```
{
  "token": "string"
}
```

Body__deactivate__token__tokens__deactivate__post

Properties

Name	Type	Required	Restrictions	Description
token	string	true	none	none

1.4.4.10.2 Body__delete__token__tokens__delete__delete

```
{
  "token": "string"
}
```

Body__delete__token__tokens__delete__delete

Properties

Name	Type	Required	Restrictions	Description
token	string	true	none	none

1.4.4.10.3 Body__delete__unwritten__tokens__writer__delete__post

```
{
  "event": "string"
}
```

Body__delete__unwritten__tokens__writer__delete__post

Properties

Name	Type	Required	Restrictions	Description
event	string	true	none	none

1.4.4.10.4 Body__update__written__tokens__writer__update__post

```
{
  "token": "string"
}
```

Body__update__written__tokens__writer__update__post

Properties

Name	Type	Required	Restrictions	Description
token	string	true	none	none

1.4.4.10.5 Body__validate__token__tokens__validate__post

```
{
  "token": "string"
}
```

Body__validate__token__tokens__validate__post

Properties

Name	Type	Required	Restrictions	Description
token	string	true	none	none

1.4.4.11 HTTPValidationError

```
{
  "detail": [
    {
      "loc": [
        "string"
      ],
      "msg": "string",
      "type": "string"
    }
  ]
}
```

HTTPValidationError

Properties

Name	Type	Required	Restrictions	Description
detail	[ValidationError]	false	none	none

1.4.4.11.1 ValidationError

```
{
  "loc": [
    "string"
  ],
  "msg": "string",
  "type": "string"
}
```

ValidationError

Properties

Name	Type	Required	Restrictions	Description
loc	[string]	true	none	none
msg	string	true	none	none
type	string	true	none	none

1.5 Token writer

Služba zodpovedná za zapisovanie tokenu na Mifare 1k tag.

Zapisovanie funguje nasledovne: - zapisovačka čaká, pokým sa Mifare tag nachádza v dosahu zapisovačky - prečíta zo štvrtého bloku zapísané 128-bitové číslo - pošle požiadavku na token manager na deaktiváciu prečítaného čísla - pošle požiadavku na token manager na vygenerovanie nového 128-bitového čísla - zapíše hodnotu na tag - verifikuje zapísanú hodnotu jej prečítaním z tagu - po úspešnej verifikácií pošle požiadavku na token manager o aktiváciu tokenu

Pre zamedzenie zapísania dvoch tokenov na jedno priloženie je vytvorený cooldown 30 sekúnd, ktorý v tejto dobe neumoží zapísať na rovnaký Mifare tag znova ďalší token.

Služba používa linuxový wrapper popísaný nižšie.

1.5.1 Linuxový wrapper pre SL600-NFC zapisovačku

Implementované funkcionality: - Vypnutie LED svetla - Zapnutie LED svetka - Čítanie z Mifare 1k tagu (štvrtý blok) - Zápis na Mifare 1k tagu (štvrtý blok) - Validácia zápisu

Ukážkový kód v knižnici zapíše náhodné 128-bitové číslo do štvrtého bloku Mifare 1k tagu, potom ho prečíta z neho a tým zvaliduje zápis. Ak všetko prebehne úspešne, program skončí.

Pre vývoj bez použitia dockera je potrebné nainštalovať závislosť pyusb:

```
pip3 install pyusb
```

Program musí bežať so sudo oprávneniami

Funguje iba na Linuxe (vo WSL 2 nekomunikuje s čítačkou). Testované na Ubuntu 20.04 a Ubuntu 22.04. ## 1.6 Voting process manager

Hlavná služba na gateway-i zodpovedná za spustenie a zastavenie volieb, registráciu volebných terminálov, poskytuje informáciu o stave pripojených terminálov a udalosti o spustení a zastavení volieb. Rovnako zabezpečuje generovanie zápisnice a odoslanie zápisnice na server.

1.6.1 Registrácia volebného terminálu

Pri spustení volebného terminálu sa terminál dopytuje na endpoint `/register-vt` kedy sa pri spustenej registrácii vymení verejný kľúč gataway-a, aby mohla prebiehať šifrovaná komunikácia medzi volebným terminálom a gateway-om. Ak registrácia nie je spustená vráti sa status `400`.

1.6.2 Komunikácia medzi volebným terminálom

Táto služba komunikuje so všetkými registrovanými volebnými terminálmi pomocou websocketov. Vo websockete sa posiela udalosť `actual_state`, ktorý obsahuje aktuálny stav volieb volebným terminálom. Rovnako aj volebné terminály notifikujú gateway o ich aktuálnom stave udalosťou `vt_status`.

1.6.3 Popis API

root___get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/voting-process-manager-api/', headers =
   headers)

print(r.json())
```

`GET /`

Root

Simple hello message.

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

election__config__election__config__get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r =
   requests.get('/gateway/voting-process-manager-api/election-config',
   headers = headers)

print(r.json())
```

`GET /election-config`

Election Config

Returns necessary config fields for gateway from config.

Example responses

200 Response

```json
{
  "status": "string",
  "texts": {}
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

terminals__status__terminals__status__get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json',
  'Authorization': 'Bearer {access-token}'
}

r =
   requests.get('/gateway/voting-process-manager-api/terminals-status',
   headers = headers)

print(r.json())
```

`GET /terminals-status`

Terminals Status

Returns necessary staus information of connected voting terminals.

Example responses

200 Response

```
{
  "status": "string",
  "registration_status": false,
  "terminals": [

  ]
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

To perform this operation, you must be authenticated by means of one of the following methods: OAuth2PasswordBearer

login__for__access__token__token__post

Code samples

```python
import requests
headers = {
  'Content-Type': 'application/x-www-form-urlencoded',
  'Accept': 'application/json'
}

r = requests.post('/gateway/voting-process-manager-api/token',
   headers = headers)

print(r.json())
```

`POST /token`

Login For Access Token

Log in user using username and password.

Body parameter

```
grant_type: string
username: string
password: string
scope: ""
client_id: string
client_secret: string
```

Parameters

Name	In	Type	Required	Description
body	body	Body_login_for_access_token_token_post	true	none

Example responses

200 Response

```
{
  "access_token": "string",
  "token_type": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Token
422	Unprocessable Entity	Validation Error	HTTPValidationError

This operation does not require authentication

current__user_current__user___get

Code samples

```
import requests
headers = {
  'Accept': 'application/json',
  'Authorization': 'Bearer {access-token}'
}

r = requests.get('/gateway/voting-process-manager-api/current-user/',
    headers = headers)

print(r.json())
```

`GET /current-user/`

Current User

Example responses

200 Response

```
{
  "username": "string",
  "disabled": true
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	User

To perform this operation, you must be authenticated by means of one of the following methods: OAuth2PasswordBearer

start__voting__process__start__post

Code samples

```
import requests
headers = {
  'Accept': 'application/json',
  'Authorization': 'Bearer {access-token}'
}

r = requests.post('/gateway/voting-process-manager-api/start',
   headers = headers)

print(r.json())
```

`POST /start`

Start Voting Process

Starts elections and notify all voting terminals.

Example responses

200 Response

```
{
  "status": "string",
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

To perform this operation, you must be authenticated by means of one of the following methods: OAuth2PasswordBearer

end__voting__process__end__post

Code samples

```
import requests
headers = {
  'Accept': 'application/json',
  'Authorization': 'Bearer {access-token}'
}

r = requests.post('/gateway/voting-process-manager-api/end', headers
    = headers)

print(r.json())
```

`POST /end`

End Voting Process

Stops elections and notify all voting terminals.

Example responses

200 Response

```
{
  "status": "string",
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

To perform this operation, you must be authenticated by means of one of the following methods: OAuth2PasswordBearer

register__vt__register__vt__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/voting-process-manager-api/register-vt',
    headers = headers)

print(r.json())
```

`POST /register-vt`

Register Vt

Register a voting terminal. Returns status 400 if registration is disabled else return status 200 with id and public key.

Body parameter

```
{
  "public_key": "string"
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body_register_vt_register_vt_post	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

gateway__events_gateway__elections__events__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json',
  'Authorization': 'Bearer {access-token}'
}

r =
   requests.get('/gateway/voting-process-manager-api/gateway-elections-events',
   headers = headers)

print(r.json())
```

`GET /gateway-elections-events`

Gateway Events

Get all elections events of start and end of elections.

Example responses

200 Response

```
{
  "status": "success",
  "events": []
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

To perform this operation, you must be authenticated by means of one of the following methods: OAuth2PasswordBearer

get__first__start__gateway__elections__events__first__start__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json',
  'Authorization': 'Bearer {access-token}'
}

r =
   requests.get('/gateway/voting-process-manager-api/gateway-elections-events/first-
   headers = headers)

print(r.json())
```

`GET /gateway-elections-events/first-start`

Get First Start

Get first start of elections.

Example responses

200 Response

```
{
  "status": "string",
  "first_start": {}
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

To perform this operation, you must be authenticated by means of one of the following methods: OAuth2PasswordBearer

get__last__end__gateway__elections__events__last__end__get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json',
  'Authorization': 'Bearer {access-token}'
}

r =
   requests.get('/gateway/voting-process-manager-api/gateway-elections-events/last-e
   headers = headers)

print(r.json())
```

`GET /gateway-elections-events/last-end`

Get Last End

Get last end of elections.

Example responses

200 Response

```json
{
  "status": "string",
  "last_end": {}
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

To perform this operation, you must be authenticated by means of one of the following methods: OAuth2PasswordBearer

generate__commission__paper__commission__paper__generate__post

Code samples

```python
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r =
   requests.post('/gateway/voting-process-manager-api/commission-paper/generate',
   headers = headers)

print(r.json())
```

`POST /commission-paper/generate`

Generate Commission Paper

Generate commission paper in pdf format encoded in base64 and store it into database.

Body parameter

```
{
  "polling_place_id": 0,
  "participated_members": [
    {
      "name": "Erik Malina",
      "agree": true
    },
    {
      "name": "Ferko Jablko",
      "agree": false
    },
    {
      "name": "Adam Jahoda",
      "agree": true
    }
  ],
  "president": {
    "name": "Samo Kiwi",
    "agree": true
  }
}
```

Parameters

Name	In	Type	Required	Description
body	body	CommissionPaper	true	none

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

get__commission__paper__commission__paper__get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r =
   requests.get('/gateway/voting-process-manager-api/commission-paper',
   headers = headers)

print(r.json())
```

`GET /commission-paper`

Get Commission Paper

Get commission paper from database encoded in base64.

Example responses

200 Response

```json
{
  "status": "string",
  "data": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

send__commission__paper__commission__paper__send__post

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r =
   requests.post('/gateway/voting-process-manager-api/commission-paper/send',
   headers = headers)

print(r.json())
```

`POST /commission-paper/send`

Send Commission Paper

Send commission paper to server.

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

Schemas

1.6.3.15.1 Body__login__for__access__token__token__post

```
{
  "grant_type": "string",
  "username": "string",
  "password": "string",
  "scope": "",
  "client_id": "string",
  "client_secret": "string"
}
```

Body__login__for__access__token__token__post

Properties

Name	Type	Required	Restrictions	Description
grant_type	string	false	none	none
username	string	true	none	none
password	string	true	none	none
scope	string	false	none	none
client_id	string	false	none	none
client_secret	string	false	none	none

1.6.3.15.2 Body__register__vt__register__vt__post

```
{
  "public_key": "string"
}
```

Body_register_vt_register_vt_post

Properties

Name	Type	Required	Restrictions	Description
public_key	string	true	none	none

1.6.3.15.3 CommissionPaper

```
{
  "polling_place_id": 0,
  "participated_members": [
    {
      "name": "Erik Malina",
      "agree": true
    },
    {
      "name": "Ferko Jablko",
      "agree": false
    },
    {
      "name": "Adam Jahoda",
      "agree": true
    }
  ],
  "president": {
    "name": "Samo Kiwi",
    "agree": true
  }
```

CommissionPaper

Properties

Name	Type	Required	Restrictions	Description
polling_place_id	integer	true	none	none
participated_members	[Member]	false	none	none
president	Member	true	none	none

1.6.3.15.4 HTTPValidationError

```
{
  "detail": [
    {
      "loc": [
        "string"
      ],
      "msg": "string",
      "type": "string"
    }
  ]
}
```

HTTPValidationError

Properties

Name	Type	Required	Restrictions	Description
detail	[ValidationError]	false	none	none

1.6.3.15.5 Member

```
{
  "name": "string",
  "agree": true
}
```

Member

Properties

Name	Type	Required	Restrictions	Description
name	string	true	none	none
agree	boolean	true	none	none

1.6.3.15.6 Token

```
{
  "access_token": "string",
  "token_type": "string"
}
```

Token

Properties

Name	Type	Required	Restrictions	Description
access_token	string	true	none	none
token_type	string	true	none	none

1.6.3.15.7 User

```
{
  "username": "string",
  "disabled": true
}
```

User

Properties

Name	Type	Required	Restrictions	Description
username	string	true	none	none
disabled	boolean	false	none	none

1.6.3.15.8 ValidationError

```
{
  "loc": [
    "string"
  ],
  "msg": "string",
  "type": "string"
}
```

ValidationError

Properties

Name	Type	Required	Restrictions	Description
loc	[string]	true	none	none
msg	string	true	none	none
type	string	true	none	none

1.7 Voting service

Služba zodpovedná za overovanie prichdádzajúceho tokenu a za prijímanie hlasu z volebného terminálu.

1.7.1 Popis API

hello___get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/gateway/voting-service-api/', headers = headers)

print(r.json())
```

`GET /`

Hello

Sample testing endpoint

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

vote__api__vote__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/voting-service-api/api/vote', headers =
    headers)

print(r.json())
```

`POST /api/vote`

Vote

Receives vote with valid token, validates the token, sotres the vote and invalidates the token.

Returns: 200: Vote was successfully stored 403: Token is invalid 409: The election is not running at the moment 422: Invalid request body

Body parameter

```
{
  "voting_terminal_id": "string",
  "payload": {
    "encrypted_message": "string",
    "encrypted_object": "string"
  }
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body__vote__api__vote__post	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

token__validity__api__token__validity__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/gateway/voting-service-api/api/token-validity',
   headers = headers)

print(r.json())
```

`POST /api/token-validity`

Token Validity

Checks if the provided token is valid.

Body parameter

```
{
  "voting_terminal_id": "string",
  "payload": {
    "encrypted_message": "string",
    "encrypted_object": "string"
  }
}
```

Parameters

Name	In	Type	Required	Description
body	body	Body__token__validity__api__token__validity__post	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
422	Unprocessable Entity	Validation Error	HTTPValidationError

Response Schema

This operation does not require authentication

Schemas

1.7.1.4.1 Body__token__validity__api__token__validity__post

```
{
  "voting_terminal_id": "string",
  "payload": {
    "encrypted_message": "string",
    "encrypted_object": "string"
  }
}
```

Body__token__validity__api__token__validity__post

Properties

Name	Type	Required	Restrictions	Description
voting__terminal__id	string	true	none	none
payload	VoteEncrypted	true	none	Attributes——— - encrypted__message: str AES encrypted message.encrypted__object: str RSA encrypted AES key and other data.

1.7.1.4.2 Body__vote__api__vote__post

```
{
  "voting_terminal_id": "string",
  "payload": {
    "encrypted_message": "string",
    "encrypted_object": "string"
  }
}
```

Body__vote__api__vote__post

Properties

Name	Type	Required	Restrictions	Description
voting_terminal_id	string	true	none	none
payload	VoteEncrypted	true	none	Attributes———- encrypted_message: str AES encrypted message.encrypted_object: str RSA encrypted AES key and other data.

1.7.1.4.3 HTTPValidationError

```
{
  "detail": [
    {
      "loc": [
        "string"
      ],
      "msg": "string",
      "type": "string"
    }
  ]
}
```

HTTPValidationError

Properties

Name	Type	Required	Restrictions	Description
detail	[ValidationError]	false	none	none

1.7.1.4.4 ValidationError

```
{
  "loc": [
    "string"
  ],
  "msg": "string",
  "type": "string"
}
```

ValidationError

Properties

Name	Type	Required	Restrictions	Description
loc	[string]	true	none	none

Name	Type	Required	Restrictions	Description
msg	string	true	none	none
type	string	true	none	none

1.7.1.4.5 VoteEncrypted

```
{
  "encrypted_message": "string",
  "encrypted_object": "string"
}
```

VoteEncrypted

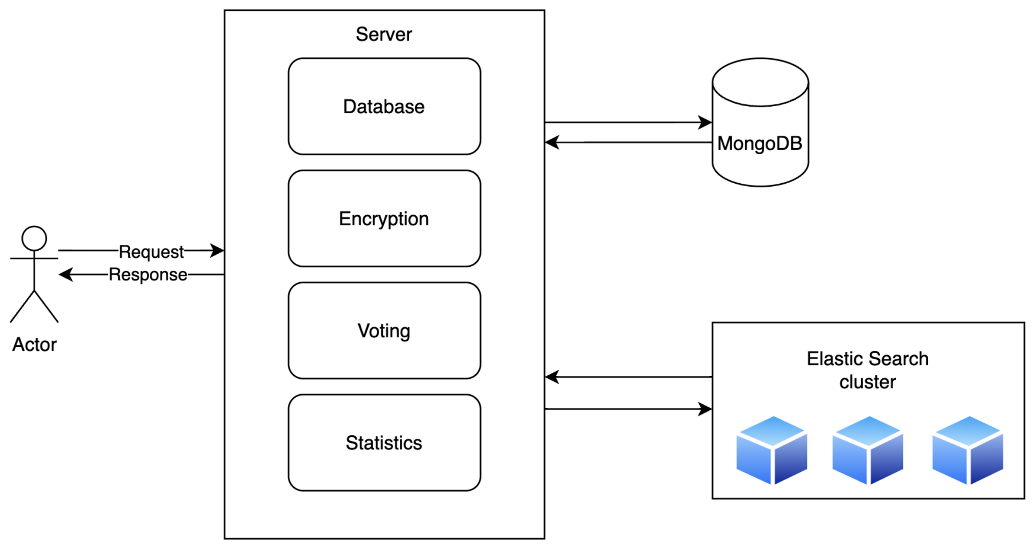
Properties

Name	Type	Required	Restrictions	Description
encrypted_message	string	true	none	none
encrypted_object	string	true	none	none

2. Server

Server je centrálna jednotka na spracovanie hlasov z volebných miestností. Server po prijatí požiadavky na uloženie hlasov zabezpečí ich validáciu, následné spracovanie a uloženie. Po úspešnom vykonaní vráti odpoveď v ktorej špeifikuje koľko hlascov bolo spracovaným. Uložené hlasy sa priebežne indexujú do technlógie Elastic Search, z ktorej sú následne získavené pri volaní koncových bodov na získanie výsledkov a štatistík.

2.1 Architektúra



2.2 Inštalácia

2.2.1 Závislosti

Pre spustenie docker kontajnerov je potrebné mať nainštalované technológie Docker, Docker compose. Pre účely vývoja ďalej odporúčame mať nainštalovaný jazyk Python, nástroj na testovanie koncových bodov ako Postman alebo Insomnia a nástroj na manipuláciu s MongoDB ako napríklad MongoDB Compass.

Knižnice pythonu su definované v textovom súbore requirements.txt, ktoré si nainštalujete príkazom: `pip install -r requirements.txt`

2.2.2 Spustenie

Lokálne samostatné spúšťanie jednotlivých častí potrebných pre chod serveru neodporúčame, z dôvodu radu problémov ktoré môžu vzniknúť. Najjednoduchším spôsobom je spustenie pomocou orchestrátora docker compose.

Prejdite do koreňového adresára servera a spustite nasledujúci príkaz.

```
docker compose up -d --build
```

Po vybudovaní by mali bežať všetky služby servera (MongoDB, FastAPI server a Elastic Search Cluster)

Zobrazenie všetkých dostupných koncových bodov servera navštívte adresu `http://localhost:8222/docs`

2.2.3 Ako si naimportovať skúšobné dáta a pripraviť Elastic Search cluster

V API docs špecifikácii spustite volania na jednotlivé koncové body v nasledovnom poradí: 1. `/database/import-data` 2. `/database/seed-votes` (s počtom hlasov, ktoré sa majú vygenerovať) 3. `/elastic/setup-elastic-vote-index` (Elastic uzly musia byť pred týmto volaním funkčné, ak nie sú, skontrolujte prosím sekciu týkajúcu sa problému s malou pamäťou dockera.) 4. `/elastic/synchronize-votes-es` (Synchronize votes in batches)

2.2.4 Problém s Elastic search pamäťou

V prípade chybovej hlášky spomínajúcej prekročenie limitu pamäte, je potrebné nastaviť premmennú vm.max_map_count v kerneli dockeru na najmenej 262144.

V závislosti od operačného systému použite jeden z nasledovných príkazov:

```
docker-machine ssh
sudo sysctl -w vm.max_map_count=262144

wsl -d docker-desktop
sysctl -w vm.max_map_count=262144
```

Na apple zariadeniach je možné toto nastavenie zmeniť priamo v nastaveniach Docker Desktop App v sekcii: Settings -> Resources -> Advanced -> Memory. 8Gb pamäte by malo postačovať.

2.2.5 Testovanie vnútri dockeru

Jednotkové testovanie vykonávané v dockeri spustíte nasledovným príkazom v priečinku zdrojových kódov servera:

```
docker-compose -p test-server -f docker-compose.test.yml up --build
   --exit-code-from server --renew-anon-volumes
```

Dostupné príznaky: - -p - preped prefix to container names - -f - docker-compose yml file - –build - build images if changed sources - –exit-code-from - get overall exit code from specified container - –force-recreate - recreate all containers - –renew-anon-volumes - delete anonym volumens

Pre zastavenie kontajnerov použite príkaz:

```
docker-compose -f docker-compose.test.yml down
```

2.3 Databáza

Server používa na ukladanie dát dokumentovú databázu MongoDB. Aj keď je do MongoDB vkladať dáta s rôznymi atribútmi, používame modely jednoitlivých dátových entít, ktoré špecifikujú štruktúru objektu a definujú typy jeho atribútov. Pracujeme s nasledujúcimi kolekciami: - votes - parties - candidates - polling_places - key_pairs

Štruktúra uloženého hlasu:

```
class Vote(BaseModel):
    token: str
    party_id: Optional[int] = None
    election_id: str
    candidate_ids: List[int] = []
```

Ďalej sa počas spracovania hlasov dynamicky pridajú dva atribúty a to:

```
polling_place_id: int
synchronized: bool
```

Atribút polling_place_id slúži na spojenie hlasu s miestnosťou, v ktorej bol zvolený a atribút synchronized, ktorý indikuje, či bol daný hlas už zandexovaný do Elastic Searchu.

Štruktúra politickej strany:

```
class Party(BaseModel):
    id: int = Field(..., alias="_id")
    party_number: int
    name: str
    official_abbr: str
    abbr: str
    image: str
    image_bytes: str
    color: str
    candidates: List[Candidate] = []
```

Dátová štruktúra politickej strany obsahuje základné údaje ako názov, skratka a číslo a doplnkové údaje ako farba a logo, ktoré sa používajú v štatistickej aplikácii. Ďalej strana obsahuje zoznam kadidátov, ktorý sú reprezentovaný vlastným modelom.

Štruktúra volebnej miestnosti:

```
class PollingPlace(BaseModel):
    id: int = Field(..., alias="_id")
    region_code: int
    region_name: str
    administrative_area_code: int
    administrative_area_name: str
    county_code: int
    county_name: str
    municipality_code: int
    municipality_name: str
    polling_place_number: int
    registered_voters_count: int
```

Dátová štruktúra volebnej miestnosti obsahuje ifnormácie o územných celkoch, v ktorých sa daná miestnosť nachádza. Tieto údaje budú následne použité na prepočítavanie výslekov pre rôzne lokality (obce, okresy a kraje).

Štruktúra kandidáta:

```
class Candidate(BaseModel):
    id: int = Field(..., alias="_id")
    party_number: int
    order: int
    first_name: str
    last_name: str
    degrees_before: str
    age: int
    occupation: str
    residence: str
```

Dátová štruktúra kandidáta obsahuje základné údaje o kandidátovy, ktoré sú použité na zobrazovanie výsledkov a obsahuje taktiež prepojenie na politickú stranu, ktorej je súčasťou.

Štruktúra kľúčového páru:

```
class KeyPair(BaseModel):
    id: int = Field(..., alias="_id")
    polling_place_id: int
    private_key_pem: str
    public_key_pem: str
    g_private_key_pem: str
    g_public_key_pem: str
```

Kľúčový pár je špecifický pre každú volebnú meistnosť a jeho privátnym kľúčom je dešifrovaná iba kominikácia, ktorá prichádza z tejto volebnej miestnosti. Tento krok zvyšuje bezpečnosť komunikácie.

2.3.1 Popis API

schema__database__schema__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/database/schema', headers = headers)

print(r.json())
```

`GET /database/schema`

Schema

Get all collections from database

Example responses

200 Response

```
{
  "collections": []
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Collections

This operation does not require authentication

import__data__database__import__data__post

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}
```

```

r = requests.post('/database/import-data', headers = headers)

print(r.json())
```

`POST /database/import-data`

Import Data

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Message

This operation does not require authentication

seed_data_database_seed_data_post

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/database/seed-data', params={
  'number_of_votes': '0'
}, headers = headers)

print(r.json())
```

`POST /database/seed-data`

Seed Data

Parameters

Name	In	Type	Required	Description
number_of_votes	query	integer	true	none

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Message
422	Unprocessable Entity	Validation Error	HTTPValidationError

This operation does not require authentication

seed__votes__database__seed__votes__post

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/database/seed-votes', params={
  'number_of_votes': '0'
}, headers = headers)

print(r.json())
```

`POST /database/seed-votes`

Seed Votes

Parameters

Name	In	Type	Required	Description
number__of__votes	query	integer	true	none

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Message
422	Unprocessable Entity	Validation Error	HTTPValidationError

This operation does not require authentication

2.4 Generovanie hlasov

Pre účely vývoja a testovania odporúčame generovať hlasy vo väčšom počte. Celý postup generovania spolu s naindexovaním prijatých hlasov dosiahnete vykonaním volaní v nasledujúcom poradí: 1. `/database/import-data` 2. `/database/seed-votes` (s počtom hlasov, ktoré sa majú vygenerovať) 3. `/elastic/setup-elastic-vote-index` (Elastic uzly musia byť pred týmto volaním funkčné, ak nie sú, skontrolujte prosím sekciu týkajúcu sa problému s malou pamäťou dockera.) 4. `/elastic/synchronize-votes-es`

V prípade potreby dogenerovania ďalších hlasov stačí vykonať kroky 2 a 4.

Ak potrebujete vymazať existujúce hlasy len z Elastic Searchu stačí spustiť krok č. 3.

V prípade potreby vymazania hlasov z MongoDB vykonajte príkazy 1 a 3.

2.5 Hlasovanie

Základná myšlienka hlasovania spočíva vo validácii prichádzajúceho zoznamu hlasov z gateway-u, ktorá musí prejsť niekoľkými krokmi. Samotný zoznam prichádzajúcich hlasov je zašifrovaný pomocou vlastnej knižnice *electiersa*, ktorého štruktúra je následovná:

```
class VoteEncrypted(BaseModel):
    encrypted_message: str
    encrypted_object: str

class VotesEncrypted(BaseModel):
    polling_place_id: int
    votes: List[VoteEncrypted] = []
```

Ak je validácia úspešná, spomínaný zoznam prichádzajúcich hlasov sa uloží do kolekcie *votes* a informuje používateľa. V opačnom prípade, server vráti špecifickú hlášku, vďaka ktorej používateľ bude vedieť, v akom kroku bola validácia neúspešná.

2.5.1 Validácia

- *id* volebnej miestnosti sa musí nachádzať v kolekcii *key_pair*
- počet kandidátov nesmie byť väčší ako 5
- každý kandidát sa v prichádzajúcom hlase môže vyskytovať iba raz
- nezadaná politická strana nesmie obsahovať žiadneho kandidáta
- kandidát musí patriť do správne politickej strany
- v kolekcii *votes* sa nesmie nachádzať duplitcitná kombinácia tokenu a *id* volebnej miestnosti
- v príchádzajúcom zozname hlasov sa nesmie nachádzať duplicitný token
- *id* volieb musí byť totožné s tým, ktoré sa nachádza v konfiguračnom súbore *config.py*

2.5.2 Popis API

vote_elections_vote_post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/elections/vote', headers = headers)

print(r.json())
```

`POST /elections/vote`

Vote

Process candidate's vote

Body parameter

```
{
  "polling_place_id": 0,
  "votes": [
    {
      "encrypted_message":
         "36AMNvcpAWdHAXKCSWexgyjxrt7xeWwhOf+oUMBqip/C051...",
      "encrypted_object":
         "1b5B/LAg2/38mot9jYzRpa9O6YwrXDilpspPrGrnTKKYUXS8..."
    }
  ]
}
```

Parameters

Name	In	Type	Required	Description
body	body	VotesEncrypted	true	none

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Message
400	Bad Request	Bad Request	Message
422	Unprocessable Entity	Validation Error	HTTPValidationError

This operation does not require authentication

get__voting__data__elections__voting__data__get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/elections/voting-data', headers = headers)

print(r.json())
```

`GET /elections/voting-data`

Get Voting Data

Downlaod voting data json using command curl http://localhost:8222/elections/voting-data > config.json

Example responses

200 Response

```json
{
  "polling_places": [],
  "parties": [],
  "key_pairs": [],
  "texts": {
    "elections_name_short": {
      "sk": "string",
      "en": "string"
    },
    "elections_name_long": {
      "sk": "string",
      "en": "string"
    },
    "election_date": {
      "sk": "string",
      "en": "string"
    }
  }
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	VotingData

This operation does not require authentication

get__zapisnica_elections__zapisnica_get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/elections/zapisnica', headers = headers)

print(r.json())
```

`GET /elections/zapisnica`

Get Zapisnica

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

2.6 Výsledky a štatistiky

Výsledky volieb sa rátajú na serveri pomocou dát získaných z Elastic Searchu a funkcie `calcualte_winning_parties_and_seats`.

Na zobrazenie výsledkov ponúkame viaceré koncové body ktoré výsledky vrátia s inou agregáciou alebo vráti len ich časť aby odpoveď nebola príliš veľká.

2.6.1 Dostupné koncové body:

- `/elastic/get-parties-results`
  - získanie výsledkov politických strán bez kandidátov
- `/elastic/get-party-candidate-results`
  - získanie výsledkov všetkých strán a kandidátov
- `/elastic/get-candidates-results`
  - získanie výsledkov všetkých kandidátov
- `/elastic/get-results-by-locality`
  - získanie výsledkov všetkých strán a kandidátov pre určitú lokalitu

2.6.2 Počítanie percent a parlamentných kresiel

Výpočet získaných kresiel sa vykonáva vo funkcii `calcualte_winning_parties_and_seats`.

```
def calcualte_winning_parties_and_seats(transformed_data):
    """
    Find parties having more than 5% (threshold) and count all votes
        for these parties.
    In case parties have less then threshold value, take all parties
    Calculate relative vote percentage from this set of parties and
        calculate result seats for each party
    """
```

Algoritmus výpočtu: 1. Prepočítať poečt získaných hlasov pre všetky strany a získať tie ktoré majú nad 5%. 2. Počet republikové číslo (počet hlasov, potrebných pre získanie jedného mandátu, ráta s pomocou čísla 151) 3. Pomocou republikového čísla určiť na koľko kresiel má strana nárok a uchovať si počet po celočíselnom delení. 4. Ak neboli rozdané všetky kreslá, tak sa doplnia postupne stranám v poradí podľa zostatku po celočíselonom delení republikovým číslom.

2.6.3 Popis API

setup_elastic_votes_index_elastic_setup_elastic_vote_index_post

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/elastic/setup-elastic-vote-index', headers =
    headers)

print(r.json())
```

`POST /elastic/setup-elastic-vote-index`

Setup Elastic Votes Index

Setup elastic search. Drop index if previously used. Create new index and variables mapping.

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Message
400	Bad Request	Bad Request	Message
500	Internal Server Error	Internal Server Error	Message

This operation does not require authentication

synchronize_votes_ES_elastic_synchronize_votes_es_post

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/elastic/synchronize-votes-es', headers = headers)

print(r.json())
```

`POST /elastic/synchronize-votes-es`

Synchronize Votes Es

Batch synchronization of votes from Mongo DB to Elastic search 3 Node cluster. Shuld be called in specific intervals during election period.

Parameters

Name	In	Type	Required	Description
number	query	any	false	none

Example responses

200 Response

```
{
  "status": "string",
  "message": "string"
}
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Message
400	Bad Request	Bad Request	Message
422	Unprocessable Entity	Validation Error	HTTPValidationError
500	Internal Server Error	Internal Server Error	Message

This operation does not require authentication

get__parties__results__elastic__get__parties__results__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/elastic/get-parties-results', headers = headers)

print(r.json())
```

`POST /elastic/get-parties-results`

Get Parties Results

Body parameter

```
{
  "party": "SME RODINA"
}
```

Parameters

Name	In	Type	Required	Description
body	body	StatisticsPerPartyRequest	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
400	Bad Request	Bad Request	Message
422	Unprocessable Entity	Validation Error	HTTPValidationError
500	Internal Server Error	Internal Server Error	Message

Response Schema

This operation does not require authentication

get__parties__with__candidates__results__elastic__get__party__candidate__results__post

Code samples

```python
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/elastic/get-party-candidate-results', headers =
    headers)

print(r.json())
```

`POST /elastic/get-party-candidate-results`

Get Parties With Candidates Results

Body parameter

```
{
  "party": "SME RODINA"
}
```

Parameters

Name	In	Type	Required	Description
body	body	StatisticsPerPartyRequest	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
400	Bad Request	Bad Request	Message
422	Unprocessable Entity	Validation Error	HTTPValidationError
500	Internal Server Error	Internal Server Error	Message

Response Schema

This operation does not require authentication

get__candidates__results__elastic__get__candidates__results__post

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.post('/elastic/get-candidates-results', headers =
   headers)

print(r.json())
```

`POST /elastic/get-candidates-results`

Get Candidates Results

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
400	Bad Request	Bad Request	Message
500	Internal Server Error	Internal Server Error	Message

Response Schema

This operation does not require authentication

get__resilts__by__locality__mongo__elastic__get__results__by__locality__mongo__get

Code samples

```
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/elastic/get-results-by-locality-mongo', headers =
   headers)

print(r.json())
```

`GET /elastic/get-results-by-locality-mongo`

Get Resilts By Locality Mongo

Used to provide benchmark for ES vs Mongo aggregation queries

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline

Response Schema

This operation does not require authentication

get__results__by__locality__elastic__get__results__by__locality__post

Code samples

```
import requests
headers = {
  'Content-Type': 'application/json',
  'Accept': 'application/json'
}

r = requests.post('/elastic/get-results-by-locality', headers =
   headers)

print(r.json())
```

`POST /elastic/get-results-by-locality`

Get Results By Locality

Body parameter

```
{
  "filter_by": "region_name",
  "filter_value": "Presovsky kraj",
}
```

Parameters

Name	In	Type	Required	Description
body	body	StatisticsPerLocalityRequest	true	none

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
400	Bad Request	Bad Request	Message
422	Unprocessable Entity	Validation Error	HTTPValidationError
500	Internal Server Error	Internal Server Error	Message

Response Schema

This operation does not require authentication

get__elections__status__elastic__elections__status__get

Code samples

```python
import requests
headers = {
  'Accept': 'application/json'
}

r = requests.get('/elastic/elections-status', headers = headers)

print(r.json())
```

`GET /elastic/elections-status`

Get Elections Status

Example responses

200 Response

```
null
```

Responses

Status	Meaning	Description	Schema
200	OK	Successful Response	Inline
400	Bad Request	Bad Request	Message
500	Internal Server Error	Internal Server Error	Message

Response Schema

This operation does not require authentication