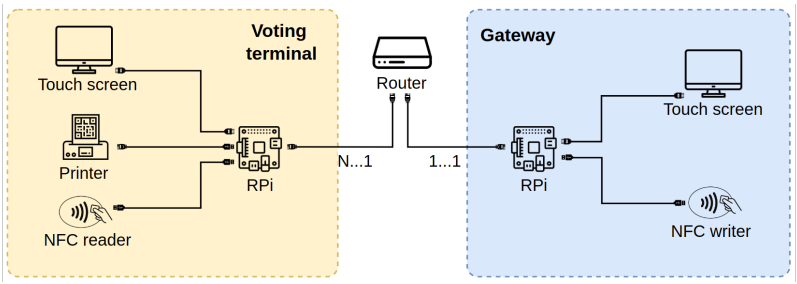
Contents

1.1 Volebný terminál

Volebný terminál je zariadenie s ktorým používateľ bezprostredne interaguje. Pozostáva z frontendu, s ktorým interaguje používateľ a backendu, ktorý obsluhuje požiadavky frontendu a komunikuje s perifémymi zariadeniami a gateway-om. Súčasťou volebného terminálu je aj čítačka NFC tagov.

1.1.1 Architektúra

Hardverové zapojenie volebného terminálu možno vidieť na nasledujúcom obrázku, pričom na obrázku je zobrazená aj architektúra gateway-a s ktorým volebný terminál komunikuje, čo je jednou z hlavných funkcionalít backendu volebného terminálu.



Z obrázku vidíme, že dotyková obrazovka s ktorou interaguje používateľ je prepojená s RPi, ktoré predstavuje základné výpočtové zariadenie celého backendu. Zariadenia sú prepojené cez HDMI. Ďalej k backendovému RPI je napojená tlačiareň pomocou sieťového kábla a čítačka NFC tagov, ktorá komunikuje s backendom cez USB rozhranie. RPi je napojené do routra volebnej miestnosti pomocou ethernetového kábla, s ktorým rovnakým spôsobom komunikuje aj gateway.

1.1.2 Mikroslužba

Ako už bolo spomenuté, volebný terminál pozostáva z frontendu a backendu, pričom je každá z týchto služieb kontajnerizovaná pomocou dockeru a spolu sú orchestrované cez docker-compose.

1.1.3 Čitačka tokenov

Najmenšou mikroslužou, ktorá je taktiež súčasťou volebného terminálu je čitačka NFC zariadenia, ktorej jedinou úlohou je prijímať vstup z čítačky a posielať obsah na endpoint v backende.

1.1.4 Frontend

Frontend volebného terminálu je implementovaný v jazyku Typescript s použitím frameworku Svetle, pričom sme pri návrhu GUI používali zaužívaný štátny dizajn manuál ID-SK.

1.1.5 Backend

Backend volebného terminálu je implentovaný v jazyku python, pričom sme použili framework FastAPI na implementáciu API rozhrania. Úlohou backendu je práve cez toto API obsluhovať požiadavky gateway-a a frontendu. V kontajneri backendu volebného terminálu je ako operačný systém použitý Alpine Linux.

Komunikácia backendu

Backend komunikuje dvoma spôsobmi a to pomocou websocketov a klasických HTTP requestov.

1.1.5.1.1 Komunikácia cez websockety

Komunikácia cez websockety je použítá na obojsmernú komunikáciu, ktorá sa nedala implementovať API rozhraním. Na implementáciu socketov sme použili knižnicu socketio. Prakticky všetky informácie ktoré posiela backend na frontend sú prenesené cez websockety. Takýmto spôsobom sa posiela na frontend napríklad aktuálny stav volieb.

Websockety sa taktiež používajú pri odosielaní volebného stavu a ID volebného terminálu na gateway. Websocketmi naopak prijíma backend informáciu o zmene stavu volieb z gateway-a.

1.1.5.1.2 Komunikácia cez API rozhranie

Komunikácia cez API rozhranie je využitá primárne na odosielanie a prijímanie hlasov. Stručne špecifikované endpointy backendu: * Endpoint na prijatie hlasu od frontendu - `/api/vote_generated` * Endpoint na prijatie tokenu - `/token` * Endpoint na prijatie stavu volieb - `/api/election/state`

Využivame aj metódu, ktorá sa spustí na začiatku životného cyklu FastAPI clienta, pričom táto metóda vykoná úvodné nastavenia volebného terminálu ako načítanie privátneho a verejného kľúča, zaregistruje volebný terminál na gateway tým, že odošle gateway-u svoj verejný kľúč a taktiež sa vytvorí websocketové pripojenie.

1.1.5.1.3 Komunikácia s tlačiarňou

Komunikácia s tlačiarňou je implementovaná cez knižnicu *CUPS*, ktorá bola do kontaineru doinštalovaná. Pri prvom odvolení sa tlačiareň zaregistruje na backende štandartným príkazom *lpadmin*. Pred samotnou registráciou tlačiarne je potrebné nainštalovať inštalačné skripty fungujúce ako driver od výrobcu tlačiarne. Z dôvodu, že používame termotlačiareň EPSON TM-T20III sú drivre verejne dostupné na adrese https://download.epson-biz.com/modules/pos/index.php?page=single_soft&cid=6918&pcat=3&pid=614
. Samotné vytlačenie hlasovacieho lístka sa potom vykoná odoslaním pdf súboru na tlač cez príkaz *lpr*.

1.1.5.1.4 Komunikácia s NFC čítačkou

NFC čítačka je HID zariadenie nad USB protokolom, teda funguje podobne ako klávesnica. Čítačka po priložení Mifare 1k tagu posiela scan kódy stlačení kláves, ktoré bolo potrebné prekonvertovať na ASCII znaky a nadpojiť na backend.

Definované metódy

Niektoré funkcionality, ktoré sú obsiahnuté v API špecifikácií sme už načrtli, okrem týchto API funkcií sú definované aj rôzne pomocné metódy, ich špecifikáciu uvádzame v tejto sekcii. Zdrojový kód je koncipovaný do piatich python súborov.

1.1.5.2.1 Súbor utils.py

- `encrypt_message(data: dict)` je metóda, ktorá pomocou knižnice na šifrovanie komunácie zašifruje dáta, ktoré dostane ako vstup
- `get_config()` je metóda, ktorá načíta súbor v JSON formáte, v ktorom sú uložené základné údaje o konfigurácii volieb
- `reg_printer()` je metóda, ktorá cez CUPS zaregistruje tlačiareň pripojenú na IP adrese definovanej v premennej prostredia
- `print_ticket_out()` je metóda, ktorá vytlačí hlasovací lístok od používateľa uložený v súbore NewTicket.pdf
- `prepare_printing_vote(vote: dict)` je metóda, ktorá vytvorí z hlasu od používateľa, ktorý je v JSON formáte PDF s obsahom hlasu
- `transform_vote_to_print(vote: dict)` je metóda, ktorá vytvorí z prijatého hlasu od používateľa slovník, kde sú namiesto čísel poslancov a strán už reálne mená a názvy

1.1.5.2.2 Súbor gateway_communication.py

- `recieve_config_from_gateway()` je metóda, ktorá cez get request natiahne a načíta configuračný súbor z gateway-a
- `send_current_election_state_to_gateway()` je metóda, ktorá odošle cez websocket volebný stav a ID volebného terminálu na gateway
- `send_token_to_gateway(token: str)` je metóda, ktorá prijme v argumente token v surovom string formáte a odošle ho na gateway, pričom v závislosti od platnosti tokenou sa aktualizuje používateľské rozhranie volebného terminálu. Ak bol token odoslaný na gateway validný, tak je možné ďalej pokračovať vo volení a stav volieb sa nastaví ako pripravený na načítanie NFC tagu
- `send_vote_to_gateway()` je metóda, ktorá v argumente dostane hlas od používateľa vo formáte slovníka a v prípade, že ešte nie je zaregistrovaná tlačiareň, tak ju zaregistruje. Po prípadnej registrácii sa hlas zašifruje a odošle na gateway. Ak bol hlas úspešne zvalidovaný, tak sa následne vytlačí hlasovací lístok.

1.1.5.2.3 Súbor main.py

V tomto súbore je uložená špecifikácia API. Okrem toho obsahuje tento súbor aj dve funkcie na komunikáciu s frontendom.

- `send_current_election_state_to_frontend()` je metóda ktorá, odošle pomocou websocketu stav volieb
- `change_state_and_send_to_frontend()` je metóda, ktorá aktualizuje stav volieb a informuje o tom aj frontend

1.1.5.2.4 Súbor imports.py

V tomto súbore sú uložené používané informácie ako ID volebného terminálu, stav volieb, informácia o registrácii tlačiarne a taktiež tento súbor obsahuje uložený prípadný zvalidovaný token s getterom a setterom.

1.1.5.2.5 Súbor NationalTicket.py

Tento súbor obsahuje triedu, ktorá predstavuje špecifikáciu volebného lístka na tlač. Trieda NationalTicket je odvodená od BaseTicket a implementuje preťažené metódy na tvorbu PDF súboru z hlasu.

- `create_pdf()` je metóda, ktorá vytvorí z hlasu PDF súbor a uloží ho do súboru NewTicket.pdf, PDF súbor sa vytvára pomocou knižnice FDPF pre Python. Funkcia na konci vytvorí z hlasu aj QR kód a vloží ho do PDF
- `preprocessText(candidates: list, max_line_len: int)` je metóda ktorá upraví hlas aby bol lepšie tlačiteľný na volebnom hlase. Úlohou metódy je dlhé mená a názvy strán vhodne rozdeliť a usporiadať
- `__init__(data: dict)` je metóda, ktorá uloži zvolený hlas do triedy, je konštruktorom triedy

1.1.6 API

hello___get

Code samples

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

r = requests.get('/backend/', 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__generated__post

Code samples

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

r = requests.post('/backend/api/vote_generated', headers = headers)

print(r.json())
```

`POST /api/vote_generated`

Vote

Api method for recieving vote from fronend

Keyword arguments: vote – vote object that user created in his action

Body parameter

```
{
  "party_id": 0,
  "candidate_ids": [
    0
  ]
}
```

Parameters

Name	In	Type	Required	Description
body	body	VotePartial	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__post

Code samples

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

r = requests.post('/backend/token', headers = headers)

print(r.json())
```

`POST /token`

Token

Api method for recieving token from client

Keyword arguments: token – token that voter user

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

receive_current_election_state_from_gateway_post

Code samples

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

r = requests.post('/backend/api/election/state', headers = headers)
```

```
print(r.json())
```

`POST /api/election/state`

Receive Current Election State From Gateway

Method for receiving current election state from gateway

Keyword arguments: state – current election state

Body parameter

```
{}
```

Parameters

Name	In	Type	Required	Description
body	body	object	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

API Schemas

HTTPValidationError

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

HTTPValidationError

1.1.6.5.1.1 Properties

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

ValidationError

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

ValidationError

1.1.6.5.1.2 Properties

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

VotePartial

```
{
  "party_id": 0,
  "candidate_ids": [
    0
  ]
}
```

VotePartial

1.1.6.5.1.3 Properties

Name	Type	Required	Restrictions	Description
party_id	integer	false	none	none
candidate_ids	[integer]	false	none	none

1.2 Dátová štruktúra

1.2.1 Štruktúra hlasu odoslaného z frontendu

Po odvolení sa odošle z frontendu hlas na backend v nasledujúcom formáte:

```
VotePartial:
      type: object
      properties:
        party_id:
          type: integer
        candidate_ids:
```

```
          type: array
          items:
            type: integer
          maxItems: 5
```

1.2.2 Štruktúra hlasu spracovaného pred tlačou

Na backende sa spracuje daný hlas vo forme slovníka. Pred tlačou je hlas transformovaný do inej podoby vďaka configuračnému súboru pre celé voľby, pričom sa *party_id* z hlasu páruje ku *_id* pre stranu v konfiguračnom súbore. Vďaka tomuto napárovaniu sa vytiahne z kofiguračného súboru meno volenej strany. Rovnakým spôsobom sa vytiahne pre každý prvok z poľa *candidate_ids* meno kandidáta. Výsledná dátová štruktúra ktorá sa posiela do funkcie na tvorbu PDF pred tlačou vyzerá nasledovne:

```
{
    "token": "valid",
    "vote": {
        "title": "ĽVoby do národnej rady",
        "candidates": ["2. Andrej Trnovec"],
        "party": "Slovenská ľudová strana Andreja Hlinku"
    }
}
```

1.2.3 Štruktúra hlasu odoslaného na uloženie na gateway

Súčasťou životného cyklu hlasu je aj odoslanie a uloženie na gateway-i. Formát finálneho hlasu odoslaného a uloženého na gateway je nasledovný:

```
Vote:
    allOf:
    - $ref: '#/components/schemas/VotePartial'
    - type: object
        required:
        - election_id
        - token
        properties:
        token:
            type: string
        election_id:
            type: string
```

Pričom *election_id* je ID volieb, ktoré práve prebiehajú.