TABLE OF CONTENTS

INTRODUCTION

This document describes the design and operational principles of a software package that implements the second factor of authentication in a corporate environment.

The first part of the document outlines the system architecture and operational logic (abstract principles and detailed algorithms).

The second part shows implementation details (DB design, screenshots for the mobile client and admin-panel, and DevOps with CI/CD).

Finally, the document includes an information security analysis: a list of potential threats that were considered and addressed during the system design stage.

Terminology used throughout the document:

Ticket	an encrypted data structure sent from the server to the client
Token (device)	a mobile device with the authentication app installed
Token (information)	a JWT token
Token activation	the process of sending a request from the token to the server

Note: All application screenshots included in the document were translated using machine translation.

1 Theoretical basis

1.1 System architecture and case scenario

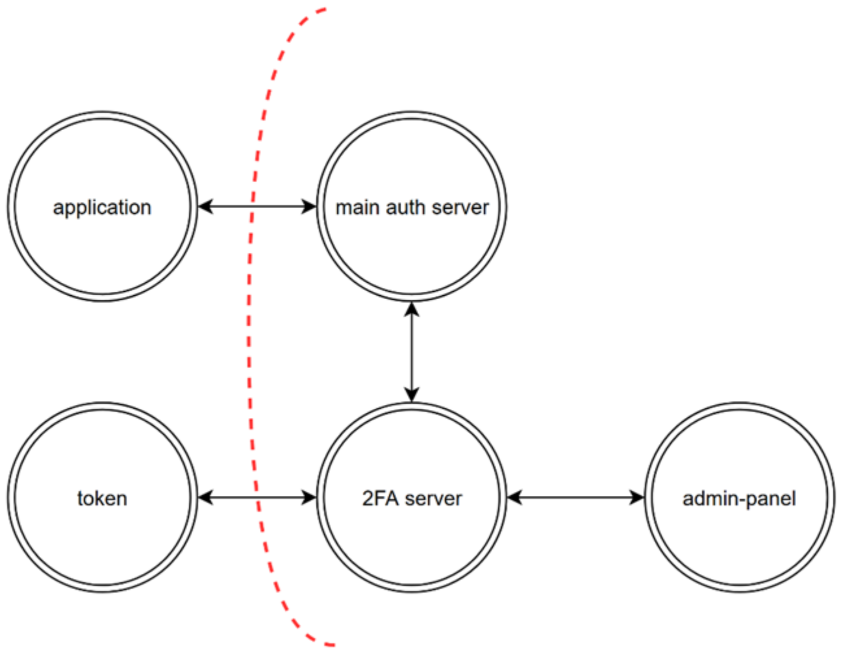
During the system architecture design, I adhered to the following principles:

– Minimizing deployment and configuration efforts — the system should be self-contained and exhibit product-like properties, making it usable without requiring in-depth knowledge of its internal operation,

– Seamless integration into existing infrastructures with minimal modifications to the primary systems (i.e., minimizing the informational footprint, specifically the amount of data stored by the first-factor server),

– Open-source distribution model, which is uncommon in corporate software (with the possibility of offering custom modifications as an optional service to customers).

The system is intended to be used in the following case scenario:



Picture 1 – 2FA process actors

The diagram above illustrates the main participants in the two-factor authentication process:

– Token – a device running the developed application.

- Application – the software component requiring authentication.
- Main auth server – a third-party system responsible for the first authentication factor.
- 2FA server – the backend for the developed system,
- Admin-panel – a tool used by system administrators to issue commands to the backend and modify its internal data.

The rationale for developing a separate admin panel is based on the practical need for enterprises to have a graphical tool that enables authorized personnel to make necessary changes to the system's behavior.

Expected communication between system components includes the following (in chronological order)

- Token → Second-factor system: the backend receives an activation request from the mobile application and responds with either a success message or an error description,
- Application → Login system: the application sends a request to initiate the identification, authentication, and authorization process,
- Login system → Second-factor system: the login system queries the second-factor backend to determine whether access is allowed for the user, receiving a yes/no response,
- Login system → Application: returns the final login status to the application (access is granted only if all authentication factors succeed).

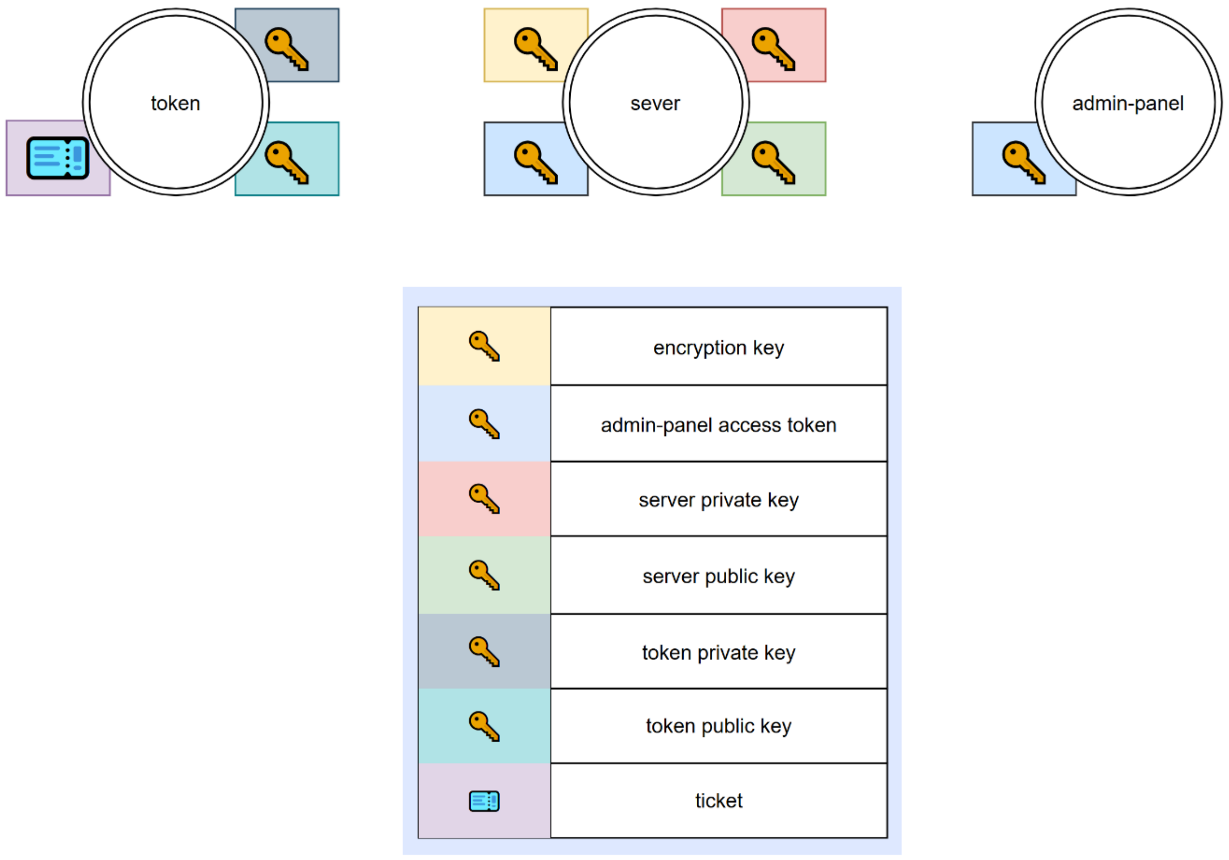
The communication between the server and the admin panel includes data retrieval and modification operations, as well as adjustments to the system configuration.

Note: "main auth server" and "application" are not part of the developed solution and are mentioned solely to illustrate the case scenario.

1.2 General principles of the application operation

1.2.1 General principles

The structure of the stored data is shown below.



🔑	encryption key
🔑	admin-panel access token
🔑	server private key
🔑	server public key
🔑	token private key
🔑	token public key
🎫	ticket

Picture 2 – Data stored by applications

The role and format of the ticket will be discussed in more detail later.

At the core of the system's functionality is the exchange of a JWT token between the client and the server.

At a high level of abstraction, communication between the server and the client can be described as follows:

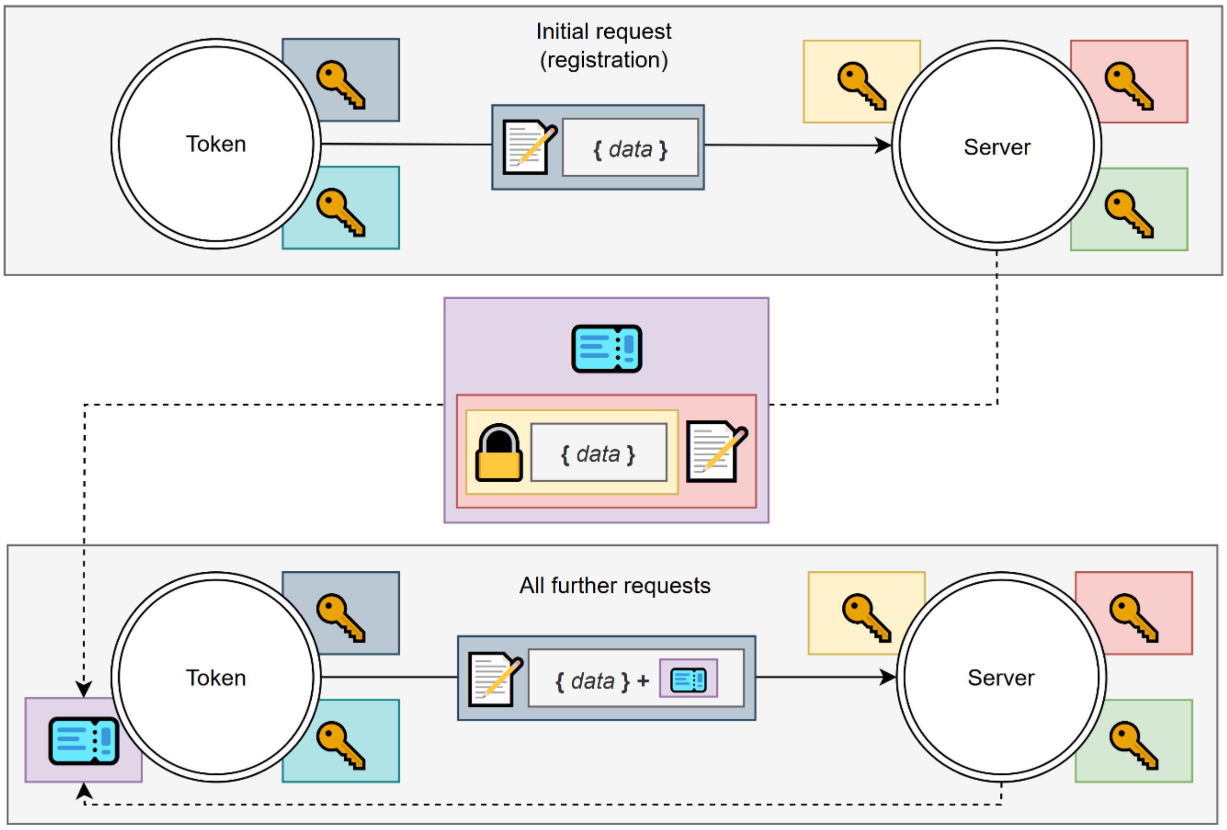
– On the initial request, the client sends a specific set of data to the server, signed with its private key (a self-signed certificate),

– The server responds with a ticket (an encrypted data block signed with the server's private key) which the client stores locally,

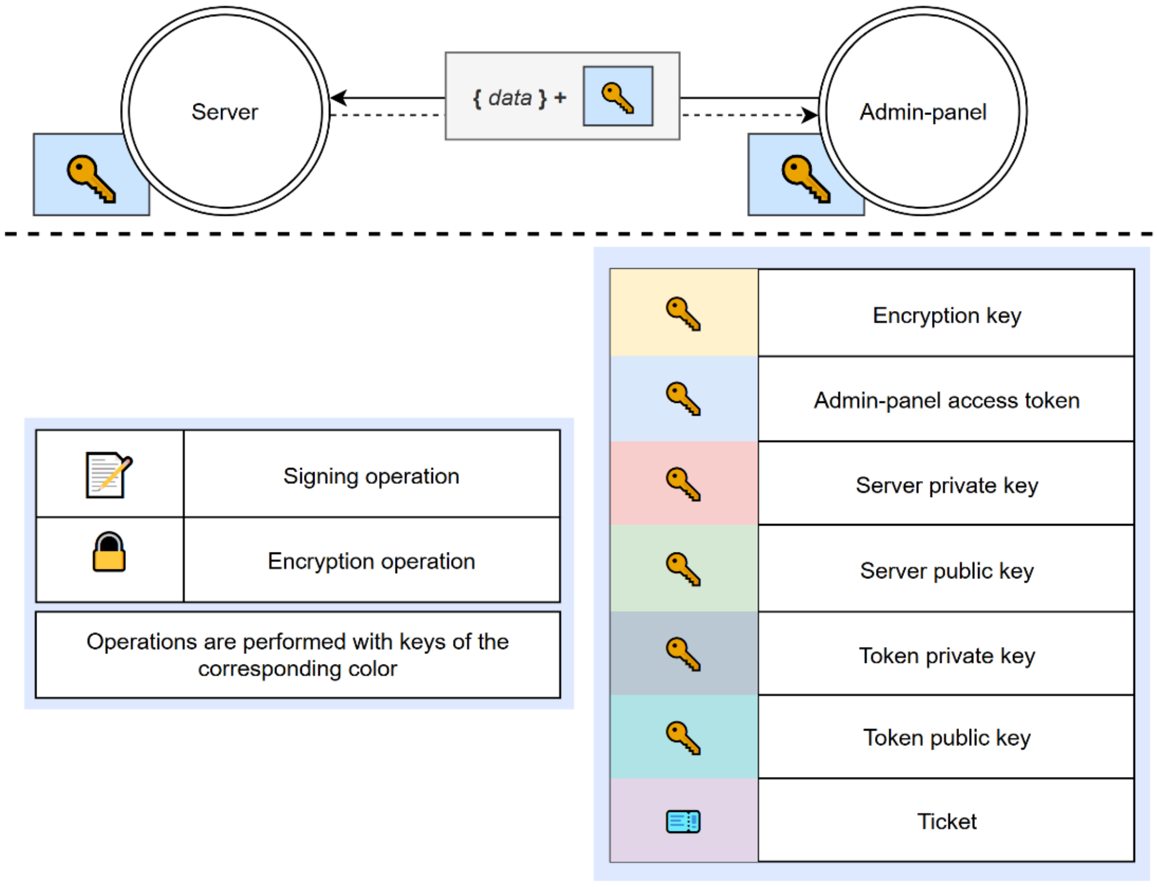
– On all subsequent requests, the client includes the most recently received ticket alongside the request data and replaces the stored ticket with the one received in the new response.

Communication between the admin panel and the server is simpler: the admin panel includes an access token as part of each request's payload.

The interaction format between the token, the server, and the admin panel is illustrated in two pictures below.



Picture 3 – Token and server interaction



Picture 4 – Admin-panel and server interaction

1.2.2 Processing request from a client

Let us now consider the general algorithm used by the system when processing a client request.

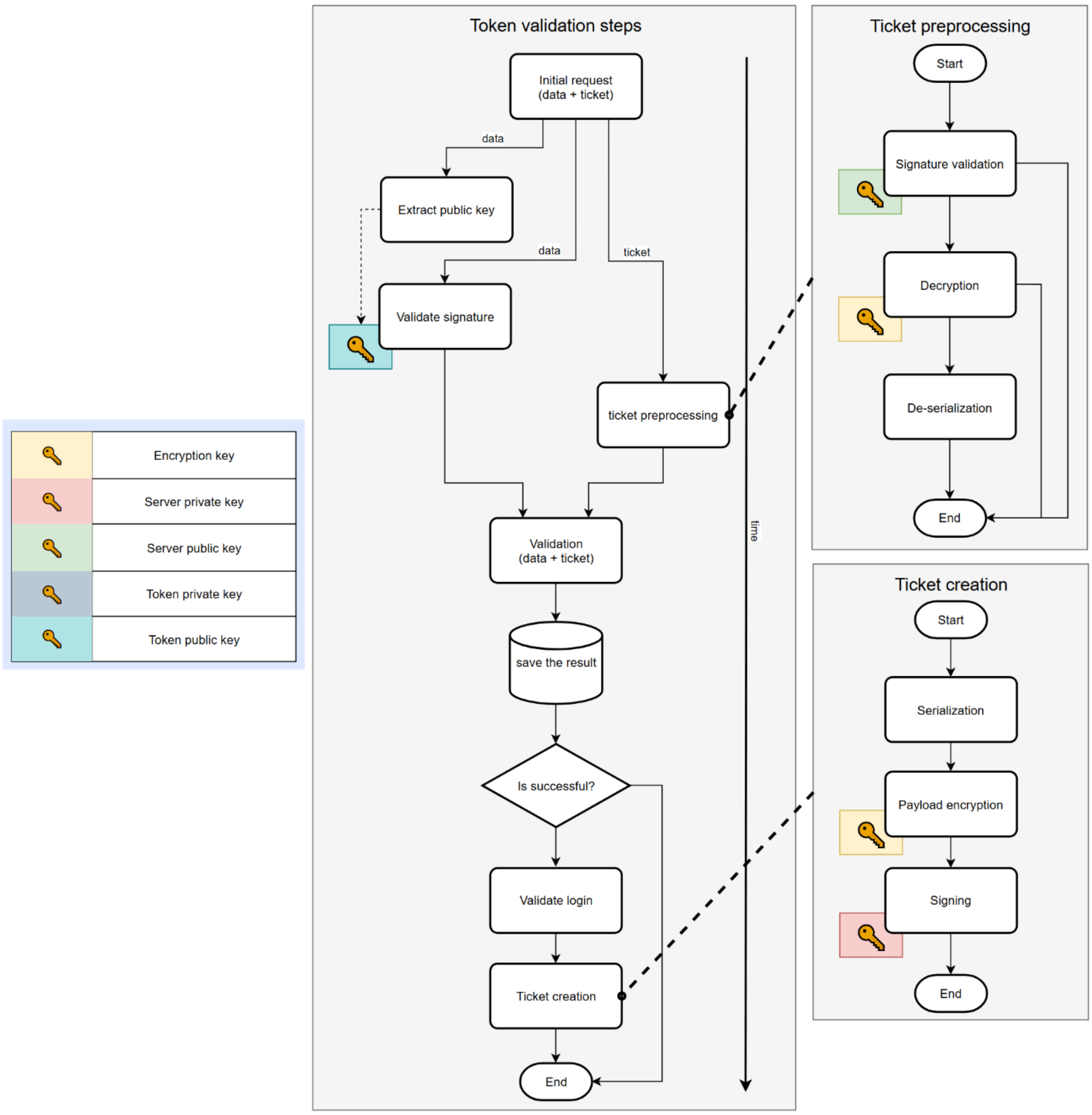
Notes on the diagram below:

– The "validate signature" and "ticket preprocessing" blocks may return an error as their output, in such cases, the "validation" block simply propagates that error further along the algorithm.

– The "validate login" block additionally updates the database with the new information or creates a new token if it does not yet exist in the system (this behavior is intentional to unify and simplify the protocol),

– During initial registration, the client does not send a ticket. In this case, "ticket preprocessing" returns an empty result, which is then handled appropriately in the "validation" block.

The general system behavior for processing a request is illustrated below



Picture 5 – System behavior when processing a request from a client

1.3 Verification algorithm and data format

1.3.1 Validation algorithm

The format of the submitted data includes the following fields.

Ticket:

– Application version for which the ticket was created,
– Ticket creation timestamp,
– Public key of the token for which the ticket was issued,

– IP address of the request for which the ticket was issued.

Token:

– Application version,
– Request timestamp,
– Public key,
– PIN code,
– Ticket (if present),
– IP address (automatically added by the server from the request data).

Some notes on the validation algorithm:

– The flowchart does not include handling of malformed data (missing fields, incorrect formats, future timestamps), such checks are present in the system but omitted from the diagram for clarity,

– Different exit points represent different error conditions,

– If the request contains a pre-existing error, the algorithm simply propagates it without further processing,

– The PIN check occurs after "Is token active?" check since an inactive token must not be allowed to reset the password.

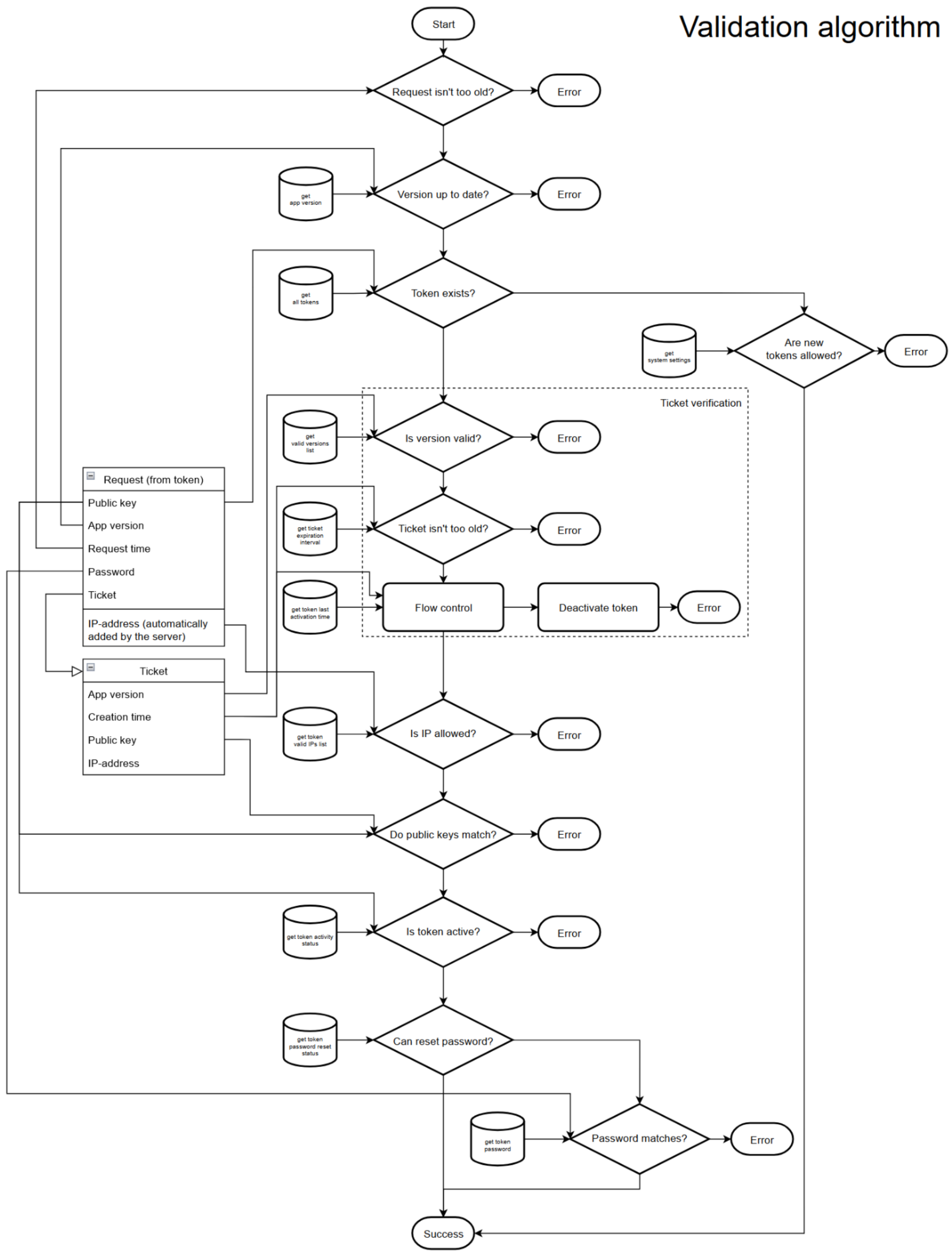
Comments on specific blocks in the algorithm:

– The initial timestamp check is used to prevent replay attacks,

– The "up to date version" and "valid version" checks differ: the first only allows the most recent version, while the second accepts a range of recent versions, as the ticket may have been generated before an update (e.g., for a previous version),

– The ticket's IP address field is currently unused and considered reserved for potential additional checks.

The flowchart illustrating the server's decision-making process when validating token activation is provided below.



Picture 6 – Token activation validation algorithm

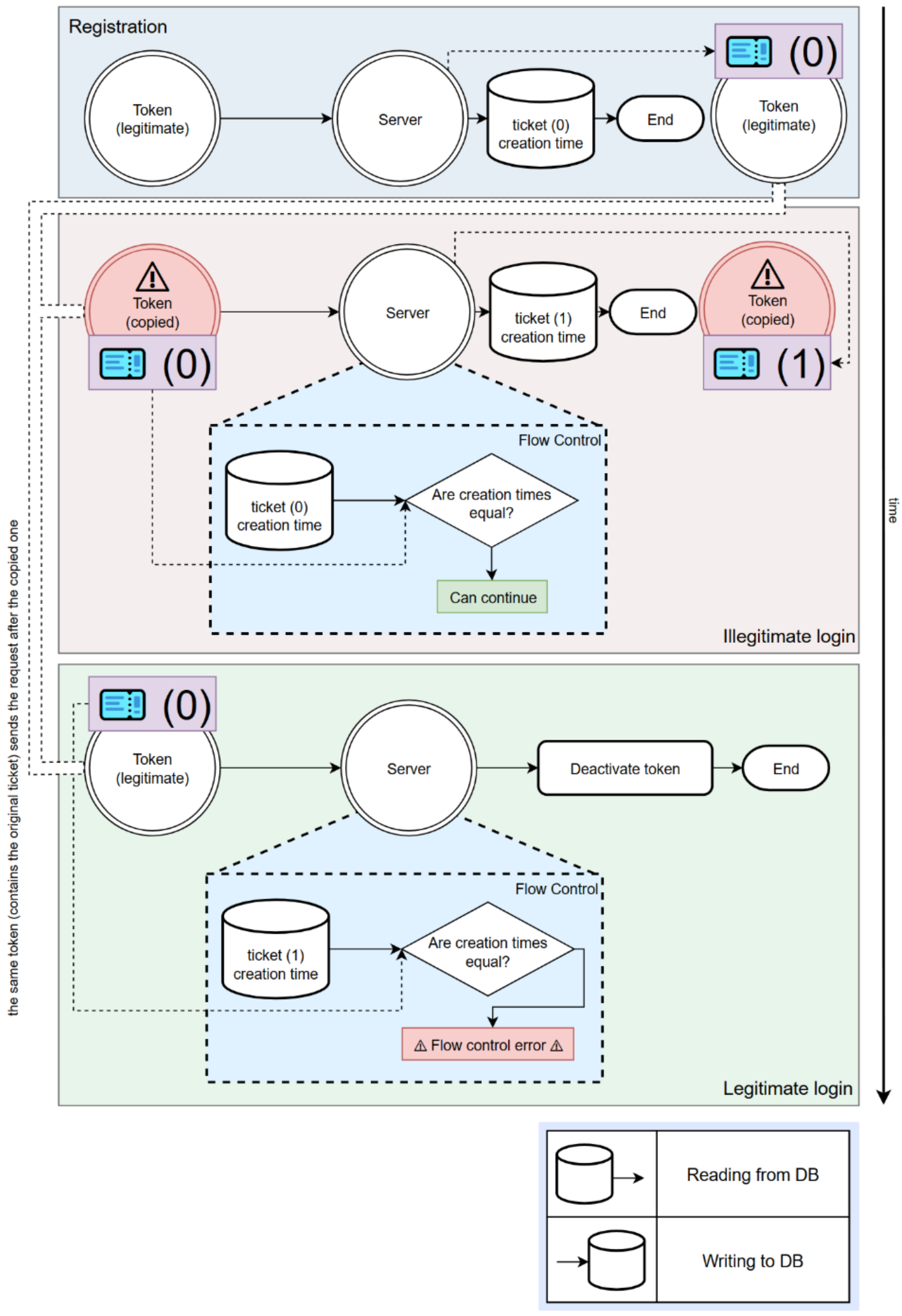
1.3.2 Flow control algorithm

The term "Flow Control" requires a separate explanation.

It functions as follows: the server compares the timestamp of the token’s last activation (which matches the ticket issuance time) with the value stored in the database. If they do not match, it is assumed that the token has been duplicated, visual demonstration is located below.

Conceptually, this check acts as a security mechanism designed to ensure that even if a token is successfully compromised, an attacker will only be able to authenticate until the legitimate user attempts to log in.

Hence the name "flow control": it refers to a sequence of token activations, where each activation requires the ticket from the previous one. Under normal conditions, this sequence is continuous (without forks or interruptions). Any disruption in the flow is considered a potential indicator of a hacking attempt.



Picture 7 – Flow Control operation diagram

1.3.3 User login request algorithm

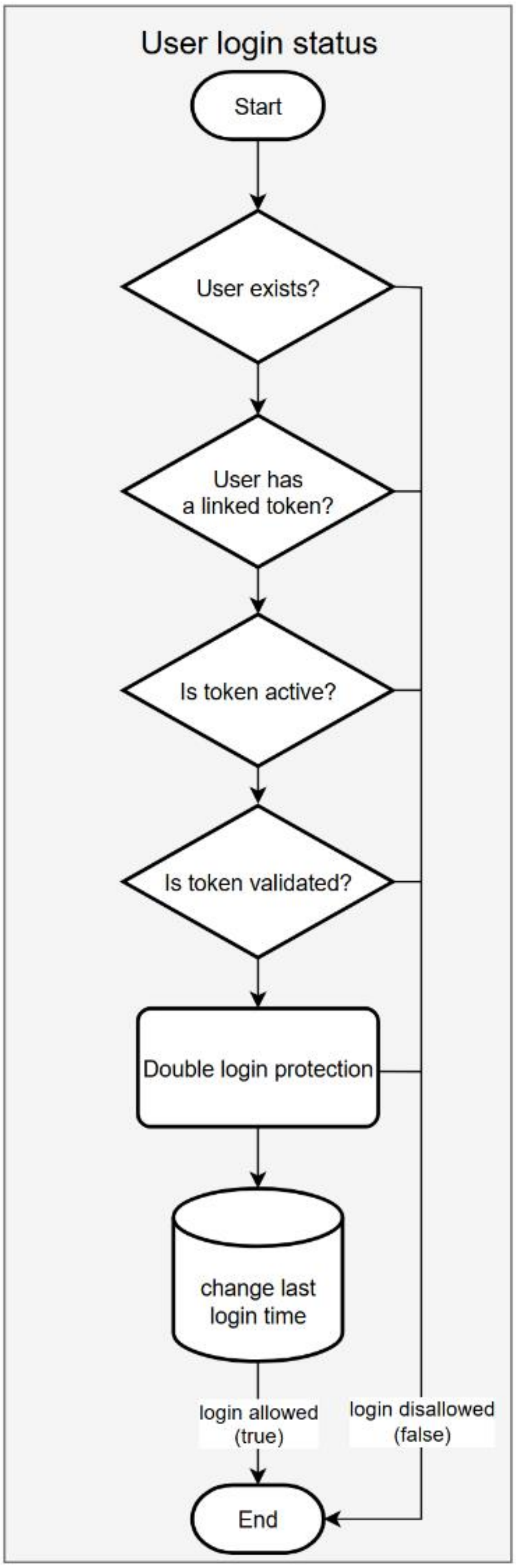
When a login request is received from the first-factor authentication system, the server returns one of two possible statuses: login allowed (true) or login denied (false). Access is granted only if none of the denial conditions are met.

Explanation of the diagram blocks:

– The check «Is token validated?» verifies whether the token’s activity window has expired,

– The double login protection» operates as follows: if the time difference between the current timestamp and the user's last login is less than the token's activity period, access is denied.

A diagram of the login decision logic is provided below.

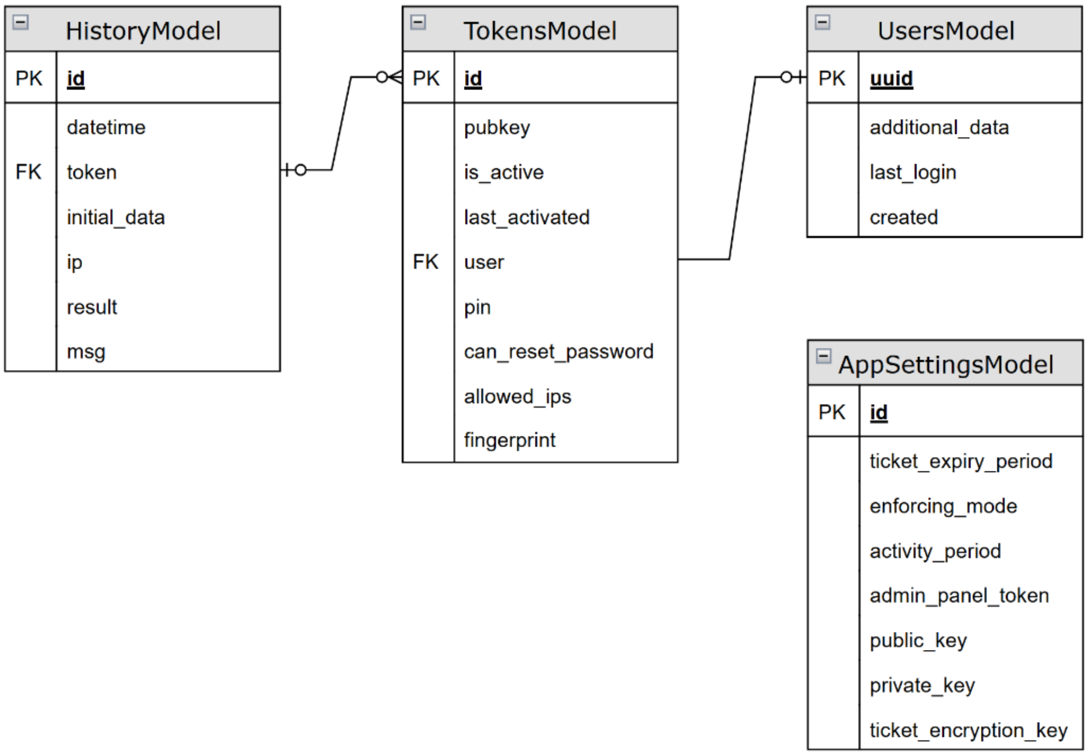


Picture 8 – User login status decision diagram

2 Practical results

2.1 DB description

The database schema is presented below (entity and field names match the actual names in the system).



Picture 9 – DB entity relation diagram

The system separates the concepts of "token" and "user": this distinction is based on their lifecycle. A token is a consumable entity with a limited lifespan, whereas a user is designed to be long-lived, as it will be referenced by external systems.

TokensModel – the primary model representing a token. Fields:

– «pubkey» – the token's public key,

– «is_active» – whether the token is currently active,

– «last_activated» – the timestamp of the token’s last activation (i.e., last ticket issuance), this is the model’s most important field, used in both flow control and login verification,

– «user» – a one-to-one relationship to the associated user, can be null,

– «pin» – a hash of the user’s password,

– «can_reset_password» – indicates whether the user is allowed to reset their password on the next login (this flag is automatically reset after each login),

– «allowed_ips» – a comma-separated list of IP addresses or subnets from which token activation is permitted, can be empty to allow activations from all IPs,

– «fingerprint» – a field derived from the public key, stored separately to allow searchability.

UsersModel – represents a user. Fields:

– «additional_data» – arbitrary text data associated with the user, may be used by system administrators for tagging,

– «last_login» – the user’s last login time, used in double login protection,

– «created» – the user creation timestamp, used for sorting.

HistoryModel – represents the request log. Fields:

– «datetime» – the timestamp of the request,

– «token» – reference to the token used in the request, nullable if the token is not in the database or an error prevented identification,

– «initial_data» – raw request data before any processing, used for manual diagnostics in case of unknown errors,

– «ip» – the IP address from which the request originated,

– «msg» – indicates whether the request completed successfully (i.e., whether the token was activated),

– «result» – error message if any, otherwise the field is empty.

AppSettingsModel – represents system configuration settings. Fields:

– «ticket_expiry_period» – the time period after which a ticket is considered expired,

– «enforcing_mode» – system mode configuration, defines whether new tokens can be created and whether existing tokens are validated (4 combinations of two binary flags),

– «activity_period» – time frame during which a confirmation from token is considered valid (used in the “Is token validated?” block when determining user login status),
– «admin_panel_token» – access token for the admin-panel,
– «public_key» – server public key,
– «private_key» – server private key,
– «ticket_encryption_key» – key used for ticket encryption.

2.2 Technologies used

The server-side application is built using the following technologies: the Python programming language, the Django web framework, and the Django Rest Framework module.

The admin panel is developed in C# as a WinForms application.

The mobile application was developed in Kotlin for the Android platform, which enabled the use of the following technologies:

– Android KeyStore API was used as a secure key storage, enabling key pair generation on a dedicated TPM module with access restricted only to biometric authentication,
– EncryptedSharedPreferences API was used as a secure ticket storage, encrypting data with a key stored in the same KeyStore.

2.3 Application description

All application screenshots are given in two versions: original Russian and machine translated into English.

2.3.1 Mobile application

During application development, I followed these key principles:

– The primary optimization of user experience is adherence to a "one-button" design philosophy, where a single button press is all that is necessary to complete the process,

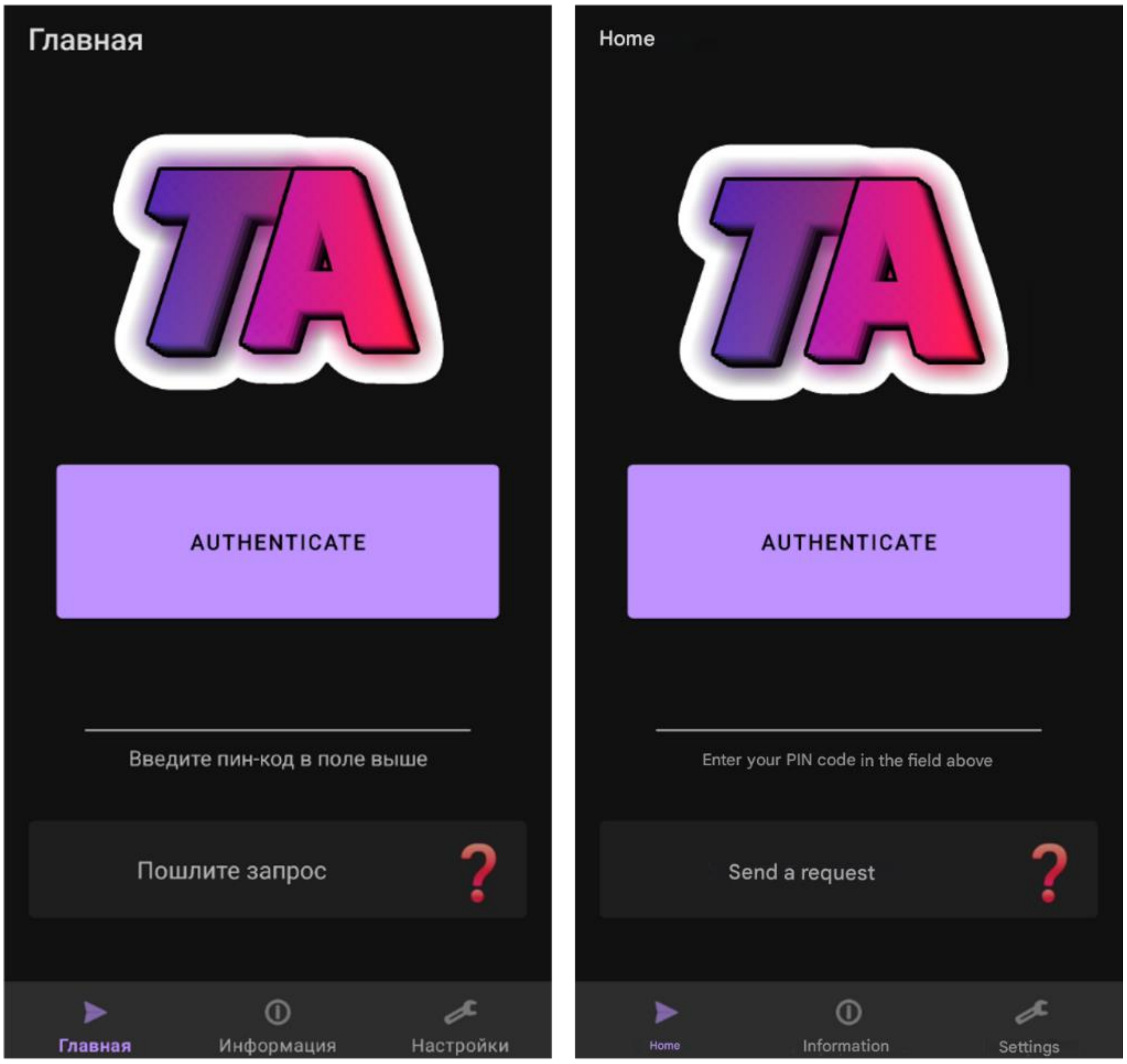
- The application must clearly indicate the specific stage at which an error occurred during token activation (this reduces unnecessary support requests and accelerates issue resolution when they do occur),
- In the case of a successful login, the application may display additional information, such as the user's login history (to help users independently detect anomalies and promptly report them to the security team),
- When used by a system administrator, the application must provide access to technical data processed by the token to facilitate root cause analysis.

Logically, the client application was divided into three screens:

- Authentication (main) screen,
- Information output screen,
- Debugging screen.

2.3.1.1 Main screen of the application

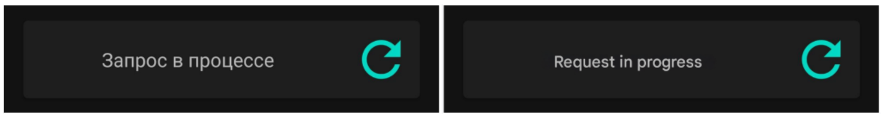
The appearance of the main screen is shown below.



Picture 10 – Mobile application home screen

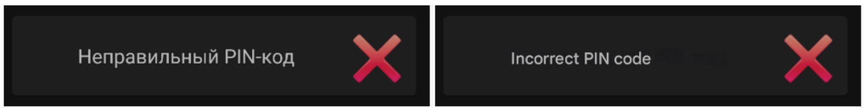
This screen also contains an application status field that dynamically displays the current activity, such as:

– Waiting for a response,



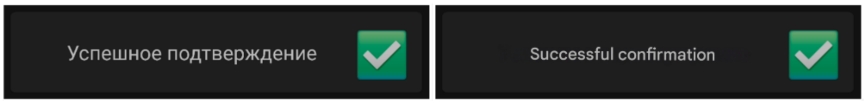
Picture 11 – Request in progress

– Error as the result of the request,



Picture 12 – Wrong PIN-code

– Or a successful confirmation.


Успешное подтверждение ✓

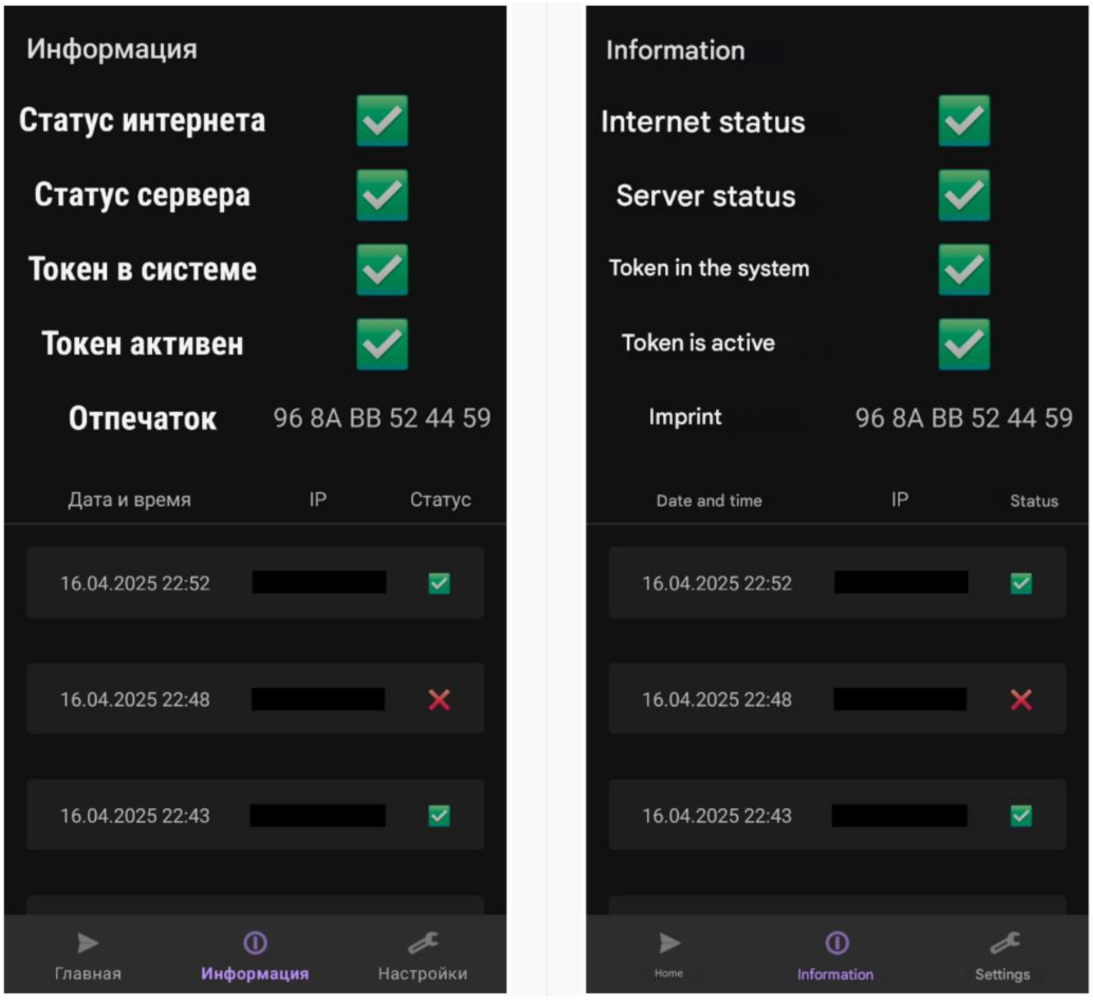
Successful confirmation ✓

Picture 13 – Successful confirmation

One of the interface elements is the use of icons that stand out in color and size. This makes it visually easier for the user to read and identify information.

2.3.1.2 Information output screen

The second screen shows the status of the last request to the user (for self-diagnostic purposes). Additionally, if the request was successful, information about the login history is displayed.


Информация

Статус интернета	✓
Статус сервера	✓
Токен в системе	✓
Токен активен	✓
Отпечаток	96 8A BB 52 44 59

Дата и время	IP	Статус
16.04.2025 22:52	[redacted]	✓
16.04.2025 22:48	[redacted]	✗
16.04.2025 22:43	[redacted]	✓

Information

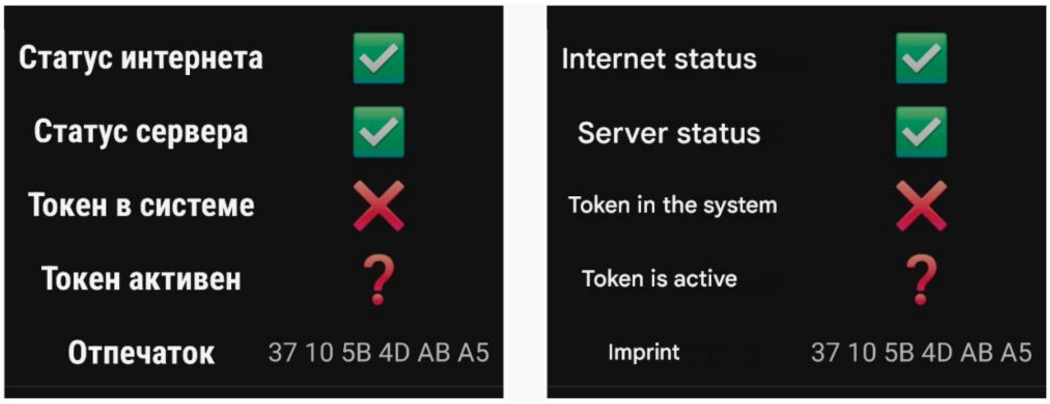
Internet status	✓
Server status	✓
Token in the system	✓
Token is active	✓
Imprint	96 8A BB 52 44 59

Date and time	IP	Status
16.04.2025 22:52	[redacted]	✓
16.04.2025 22:48	[redacted]	✗
16.04.2025 22:43	[redacted]	✓

Picture 14 – Information screen

On this screen, the user can visually identify at what stage an error occurred during the processing of a request. Here the step-by-step error display approach becomes obvious.

Instead of a just displaying the “token not found” text, which can be confusing to the untrained user, the system visually indicates at which step in the sequence an error occurred, and the processing of the request was interrupted.

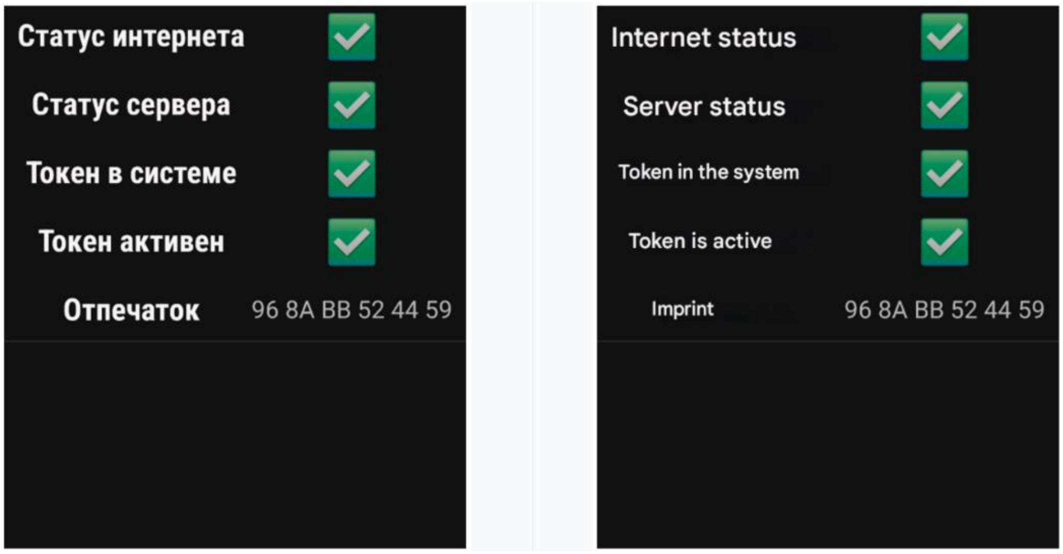


Статус интернета	✓
Статус сервера	✓
Токен в системе	✗
Токен активен	?
Отпечаток	37 10 5B 4D AB A5

Internet status	✓
Server status	✓
Token in the system	✗
Token is active	?
Imprint	37 10 5B 4D AB A5

Picture 15 – “Token not found” error

If the request ended with a user error rather than a system error, the screen would look like this (there is no login history, because it is returned only when the token is successfully activated).



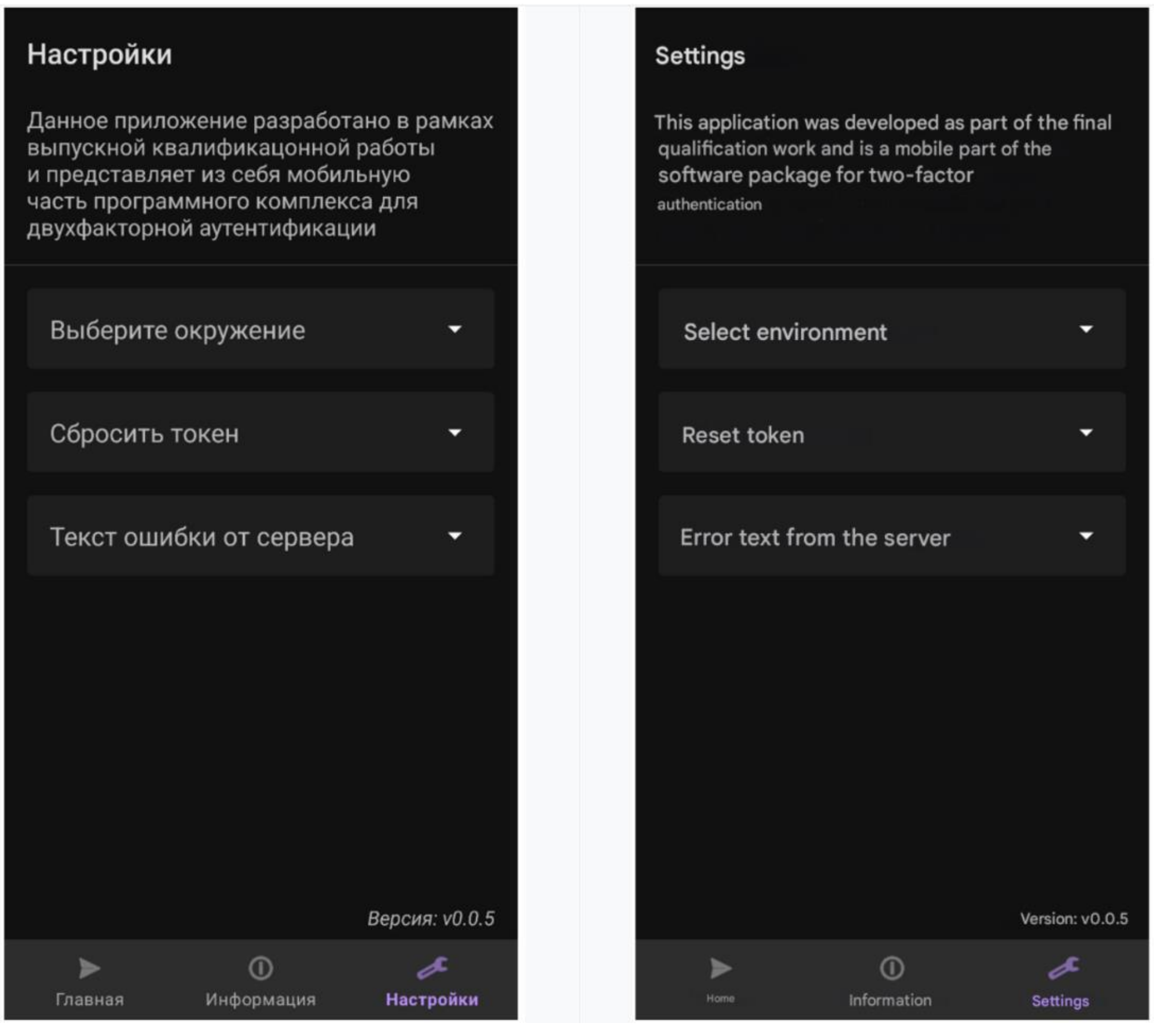
Статус интернета	✓
Статус сервера	✓
Токен в системе	✓
Токен активен	✓
Отпечаток	96 8A BB 52 44 59

Internet status	✓
Server status	✓
Token in the system	✓
Token is active	✓
Imprint	96 8A BB 52 44 59

Picture 16 – Second screen with no error

2.3.1.3 Debugging screen

The last part of the application is a window where you can access service information. It consists of expanding submenu items, each of which provides a separate block of information about the application.



Настройки

Данное приложение разработано в рамках выпускной квалификацонной работы и представляет из себя мобильную часть программного комплекса для двухфакторной аутентификации

Выберите окружение

Сбросить токен

Текст ошибки от сервера

Версия: v0.0.5

Главная Информация **Настройки**

Settings

This application was developed as part of the final qualification work and is a mobile part of the software package for two-factor authentication

Select environment

Reset token

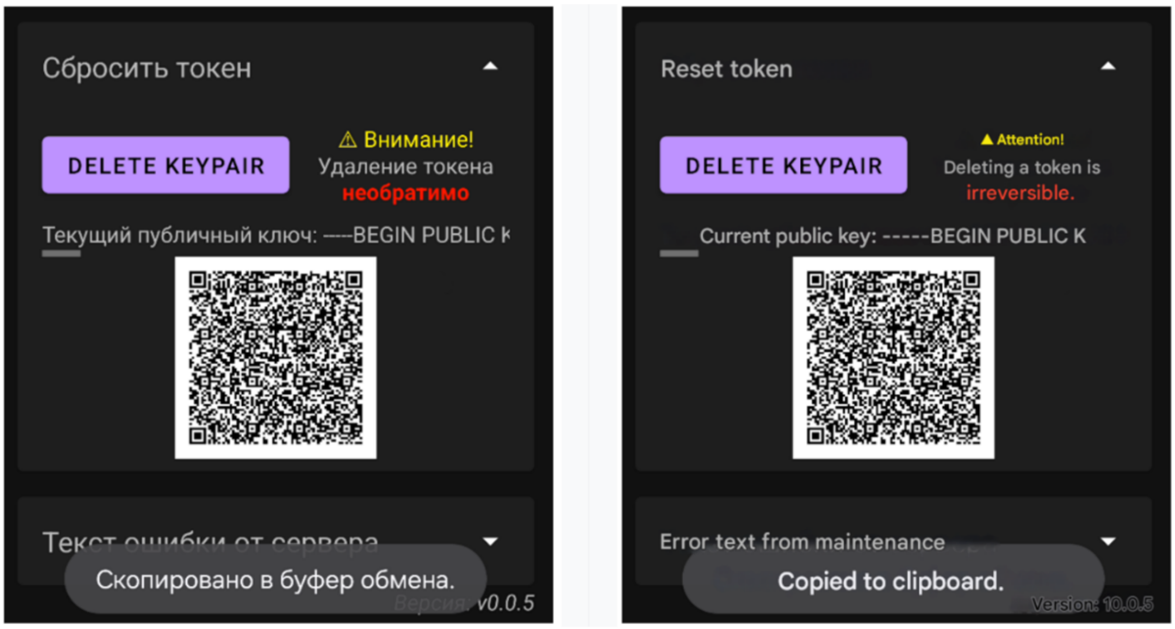
Error text from the server

Version: v0.0.5

Home Information **Settings**

Picture 17 – Settings screen

The token reset menu additionally gives access to the public key (as a scrollable and copyable text box: either by clicking or via a displayed QR code that the technician can scan on their device).



⚠ Внимание! Удаление токена необратимо

Текущий публичный ключ: ----BEGIN PUBLIC K

Скопировано в буфер обмена.

Версия: v0.0.5

▲ Attention! Deleting a token is irreversible.

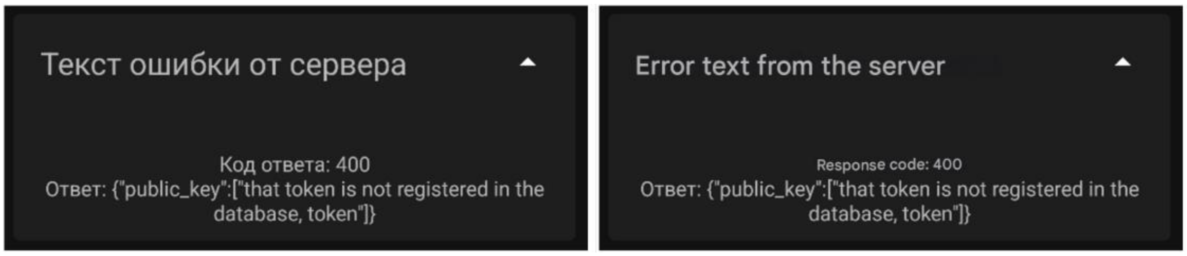
Current public key: -----BEGIN PUBLIC K

Copied to clipboard.

Version: 10.0.5

Picture 18 – Token resetting submenu

Next, the last submenu contains the response from the server in raw form. The output of this information was added to debug errors that the application could not output by standard means.

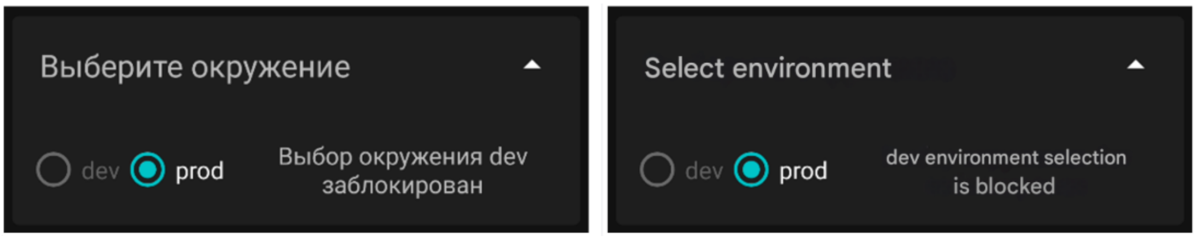


```
Код ответа: 400
Ответ: {"public_key":["that token is not registered in the database, token"]}
```

```
Response code: 400
Ответ: {"public_key":["that token is not registered in the database, token"]}
```

Picture 19 – Error text from the server

Finally, the very first menu contains the environment selection functionality.



dev [✓ prod]

Выбор окружения dev заблокирован

dev [✓ prod]

dev environment selection is blocked

Picture 20 – Environment selection

2.3.2 Admin-panel

When developing the admin panel, I designed it around the following potential usage scenarios:

– monitoring (real-time access to up-to-date information about login history),

– technical support (mainly consisting of modifying individual token fields),

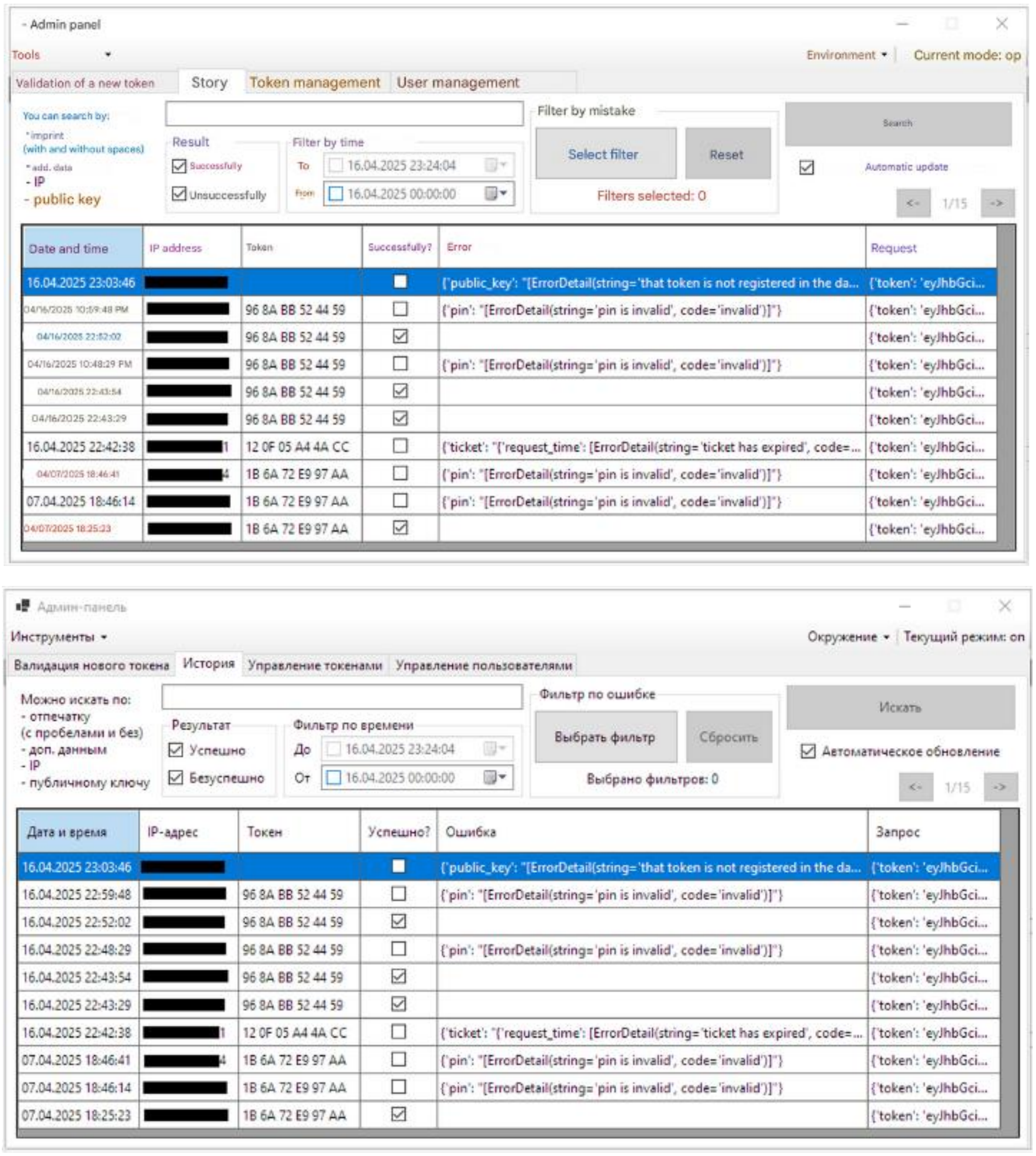
– initial setup of an individual token (the panel lets admin perform all actions necessary for the second factor for a new user to be fully ready for operation).

The application contains four main screens, each responsible for one of the three scenarios discussed above (technical support covers both user and token configuration windows).

2.3.2.1 History output window

The appearance of the history output window is shown below.

Visually, this window and the others consist of a block of filter and search settings, and a table with the results below it.



You can search by:
*imprint
(with and without spaces)
*add. data
- IP
- public key

Result
☑ Successfully
☑ Unsuccessfully

Filter by time
To 16.04.2025 23:24:04
From 16.04.2025 00:00:00

Date and time	IP address	Token	Successfully?	Error	Request
16.04.2025 23:03:46	[redacted]		☐	{'public_key': "[ErrorDetail(string='that token is not registered in the da...	{'token': 'eyJhbGci...
04/16/2025 10:59:48 PM	[redacted]	96 8A BB 52 44 59	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
04/16/2025 22:52:02	[redacted]	96 8A BB 52 44 59	☑		{'token': 'eyJhbGci...
04/16/2025 10:48:29 PM	[redacted]	96 8A BB 52 44 59	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
04/16/2025 22:43:54	[redacted]	96 8A BB 52 44 59	☑		{'token': 'eyJhbGci...
04/16/2025 22:43:29	[redacted]	96 8A BB 52 44 59	☑		{'token': 'eyJhbGci...
16.04.2025 22:42:38	[redacted]1	12 0F 05 A4 4A CC	☐	{'ticket': "{'request_time': [ErrorDetail(string='ticket has expired', code=...	{'token': 'eyJhbGci...
04/07/2025 18:46:41	[redacted]4	1B 6A 72 E9 97 AA	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
07.04.2025 18:46:14	[redacted]	1B 6A 72 E9 97 AA	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
04/07/2025 18:25:23	[redacted]	1B 6A 72 E9 97 AA	☑		{'token': 'eyJhbGci...

Можно искать по:
- отпечатку
(с пробелами и без)
- доп. данным
- IP
- публичному ключу

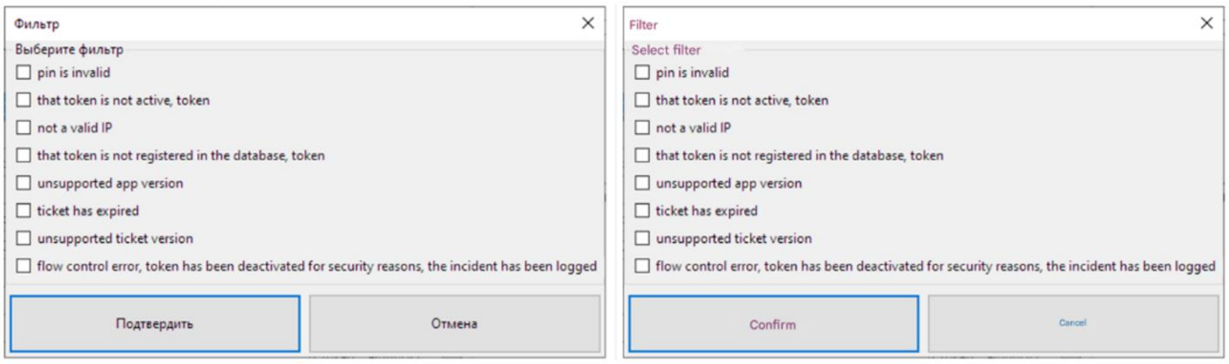
Результат
☑ Успешно
☑ Безуспешно

Фильтр по времени
До 16.04.2025 23:24:04
От 16.04.2025 00:00:00

Дата и время	IP-адрес	Токен	Успешно?	Ошибка	Запрос
16.04.2025 23:03:46	[redacted]		☐	{'public_key': "[ErrorDetail(string='that token is not registered in the da...	{'token': 'eyJhbGci...
16.04.2025 22:59:48	[redacted]	96 8A BB 52 44 59	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
16.04.2025 22:52:02	[redacted]	96 8A BB 52 44 59	☑		{'token': 'eyJhbGci...
16.04.2025 22:48:29	[redacted]	96 8A BB 52 44 59	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
16.04.2025 22:43:54	[redacted]	96 8A BB 52 44 59	☑		{'token': 'eyJhbGci...
16.04.2025 22:43:29	[redacted]	96 8A BB 52 44 59	☑		{'token': 'eyJhbGci...
16.04.2025 22:42:38	[redacted]1	12 0F 05 A4 4A CC	☐	{'ticket': "{'request_time': [ErrorDetail(string='ticket has expired', code=...	{'token': 'eyJhbGci...
07.04.2025 18:46:41	[redacted]4	1B 6A 72 E9 97 AA	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
07.04.2025 18:46:14	[redacted]	1B 6A 72 E9 97 AA	☐	{'pin': "[ErrorDetail(string='pin is invalid', code='invalid')]"}	{'token': 'eyJhbGci...
07.04.2025 18:25:23	[redacted]	1B 6A 72 E9 97 AA	☑		{'token': 'eyJhbGci...

Picture 21 – History display window

The filter window is dynamically loaded from the server and allows only certain errors to be displayed.



Выберите фильтр
☐ pin is invalid
☐ that token is not active, token
☐ not a valid IP
☐ that token is not registered in the database, token
☐ unsupported app version
☐ ticket has expired
☐ unsupported ticket version
☐ flow control error, token has been deactivated for security reasons, the incident has been logged

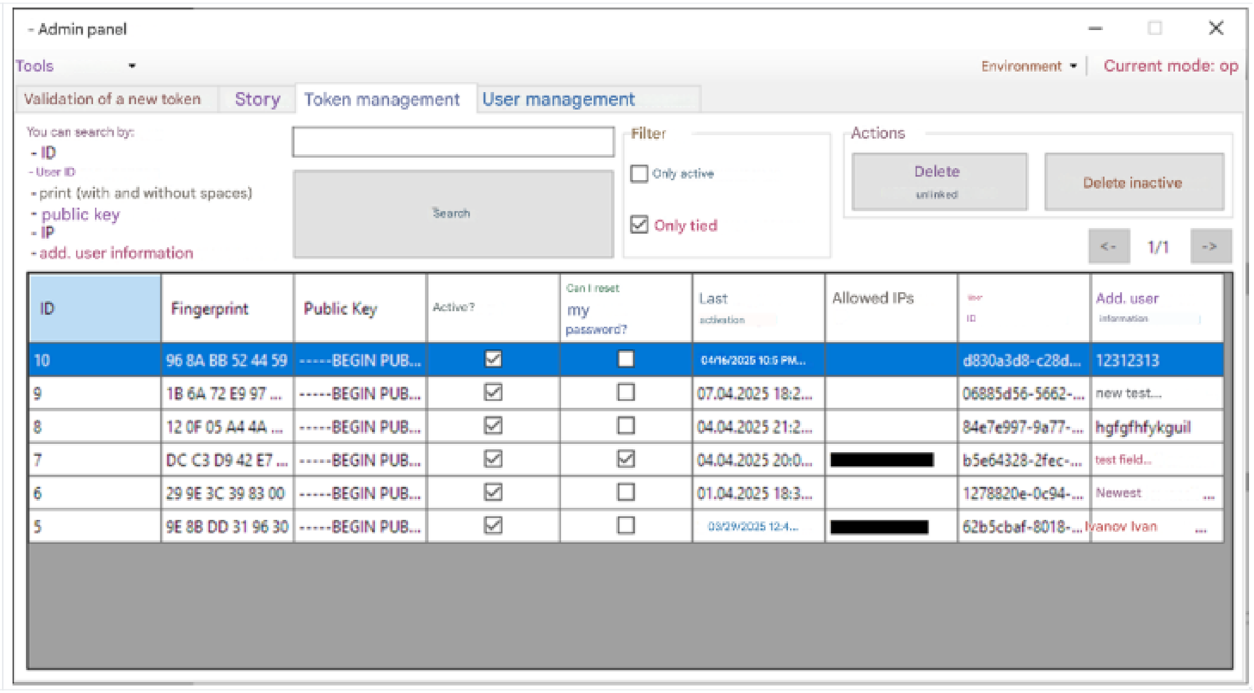
Select filter
☐ pin is invalid
☐ that token is not active, token
☐ not a valid IP
☐ that token is not registered in the database, token
☐ unsupported app version
☐ ticket has expired
☐ unsupported ticket version
☐ flow control error, token has been deactivated for security reasons, the incident has been logged

Picture 22 – Filter selection

The main difference from a user application is the absence of data preprocessing. System administrators are expected to have the skill of reading JSON objects.

2.3.2.2 Token management window

The token management window interface is shown below.

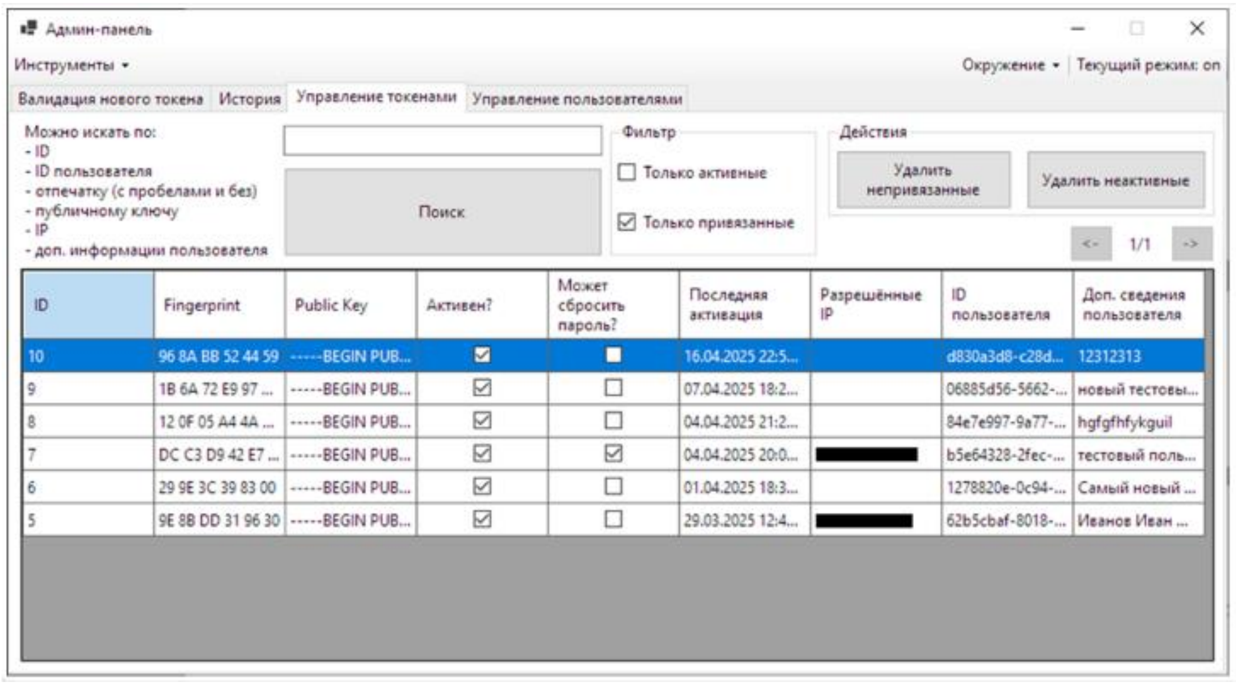


You can search by:
- ID
- User ID
- print (with and without spaces)
- public key
- IP
- add. user information

Filter: ☐ Only active ☑ Only tied

Actions

ID	Fingerprint	Public Key	Active?	Can I reset my password?	Last activation	Allowed IPs	User ID	Add. user information
10	96 8A BB 52 44 59	-----BEGIN PUB...	☑	☐	04/16/2025 10:5 PM...		d830a3d8-c28d...	12312313
9	1B 6A 72 E9 97 ...	-----BEGIN PUB...	☑	☐	07.04.2025 18:2...		06885d56-5662-...	new test...
8	12 0F 05 A4 4A ...	-----BEGIN PUB...	☑	☐	04.04.2025 21:2...		84e7e997-9a77-...	hgfgfhfykguil
7	DC C3 D9 42 E7 ...	-----BEGIN PUB...	☑	☑	04.04.2025 20:0...	[redacted]	b5e64328-2fec-...	test field...
6	29 9E 3C 39 83 00	-----BEGIN PUB...	☑	☐	01.04.2025 18:3...		1278820e-0c94-...	Newest ...
5	9E 8B DD 31 96 30	-----BEGIN PUB...	☑	☐	03/29/2025 12:4...	[redacted]	62b5cbaf-8018-...	Ivanov Ivan ...



Можно искать по:
- ID
- ID пользователя
- отпечатку (с пробелами и без)
- публичному ключу
- IP
- доп. информации пользователя

Фильтр: ☐ Только активные ☑ Только привязанные

Действия

ID	Fingerprint	Public Key	Активен?	Может сбросить пароль?	Последняя активация	Разрешённые IP	ID пользователя	Доп. сведения пользователя
10	96 8A BB 52 44 59	-----BEGIN PUB...	☑	☐	16.04.2025 22:5...		d830a3d8-c28d...	12312313
9	1B 6A 72 E9 97 ...	-----BEGIN PUB...	☑	☐	07.04.2025 18:2...		06885d56-5662-...	новый тестовы...
8	12 0F 05 A4 4A ...	-----BEGIN PUB...	☑	☐	04.04.2025 21:2...		84e7e997-9a77-...	hgfgfhfykguil
7	DC C3 D9 42 E7 ...	-----BEGIN PUB...	☑	☑	04.04.2025 20:0...	[redacted]	b5e64328-2fec-...	тестовый поль...
6	29 9E 3C 39 83 00	-----BEGIN PUB...	☑	☐	01.04.2025 18:3...		1278820e-0c94-...	Самый новый ...
5	9E 8B DD 31 96 30	-----BEGIN PUB...	☑	☐	29.03.2025 12:4...	[redacted]	62b5cbaf-8018-...	Иванов Иван ...

Picture 23 – Token management window

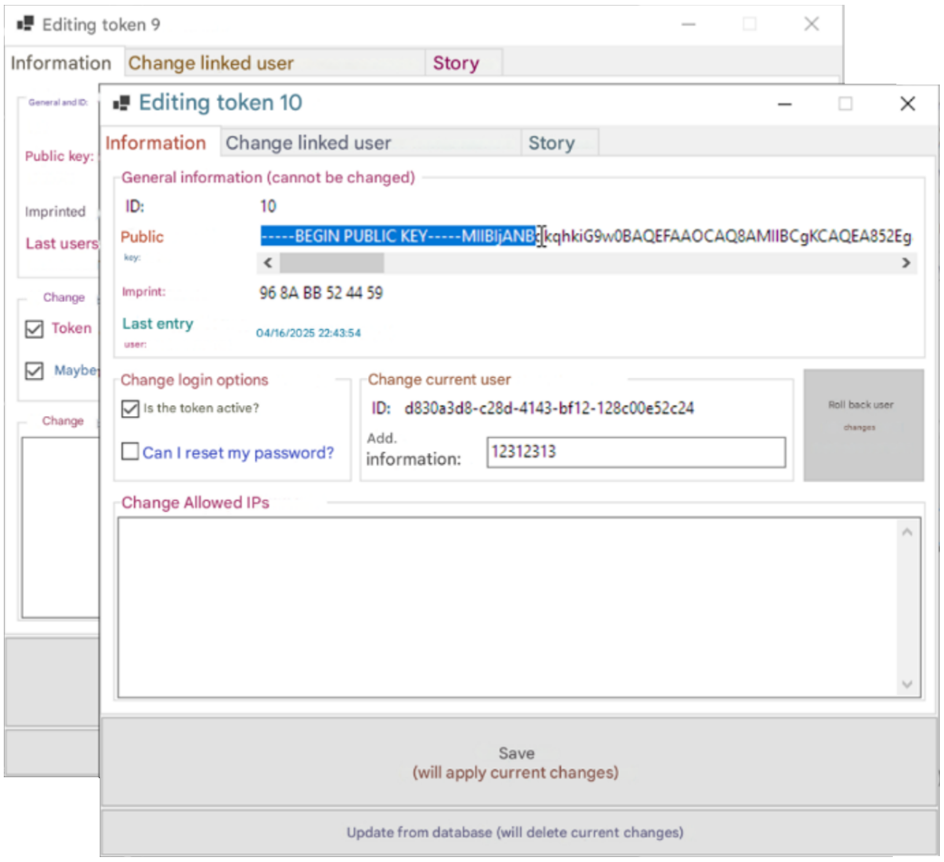
In addition to flexible searches by both token and user fields, along with the ability to display only active and/or only linked tokens (the latter is enabled by default), there are also actions that can be performed on sets of tokens.

Since tokens are considered as “consumables”, this is reflected in the deletion policy. Through the admin panel, it is only possible to bulk delete all inactive tokens and all tokens not linked to users, without the option for individual deletion.

2.3.2.3 Token modification window

The token's main information screen consists of several blocks:

- display of non-editable information (static fields),
- settings editing block (login parameters and permitted IP addresses and subnets).



Editing token 10

Information | Change linked user | Story

General information (cannot be changed)

ID: 10

Public key: -----BEGIN PUBLIC KEY-----MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA852Eg

Imprint: 96 8A BB 52 44 59

Last entry user: 04/16/2025 22:43:54

Change login options

☑ Is the token active?

☐ Can I reset my password?

Change current user

ID: d830a3d8-c28d-4143-bf12-128c00e52c24

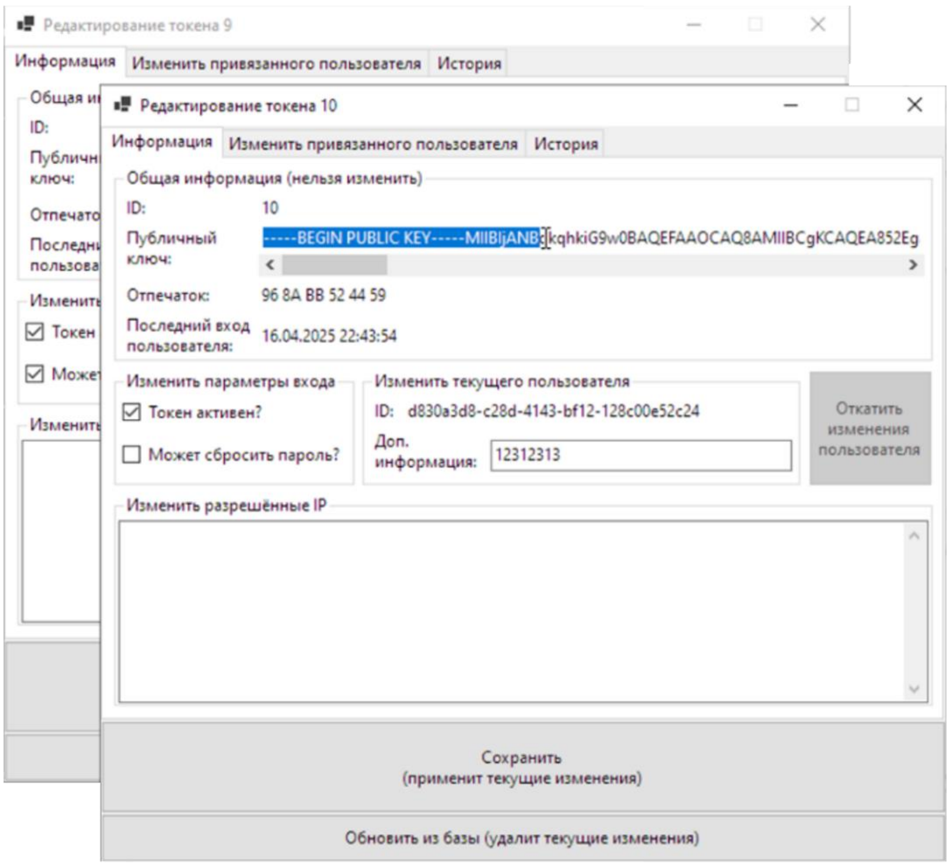
Add. information: 12312313

Roll back user changes

Change Allowed IPs

Save
(will apply current changes)

Update from database (will delete current changes)



Редактирование токена 10

Информация | Изменить привязанного пользователя | История

Общая информация (нельзя изменить)

ID: 10

Публичный ключ: -----BEGIN PUBLIC KEY-----MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA852Eg

Отпечаток: 96 8A BB 52 44 59

Последний вход пользователя: 16.04.2025 22:43:54

Изменить параметры входа

☑ Токен активен?

☐ Может сбросить пароль?

Изменить текущего пользователя

ID: d830a3d8-c28d-4143-bf12-128c00e52c24

Доп. информация: 12312313

Откатить изменения пользователя

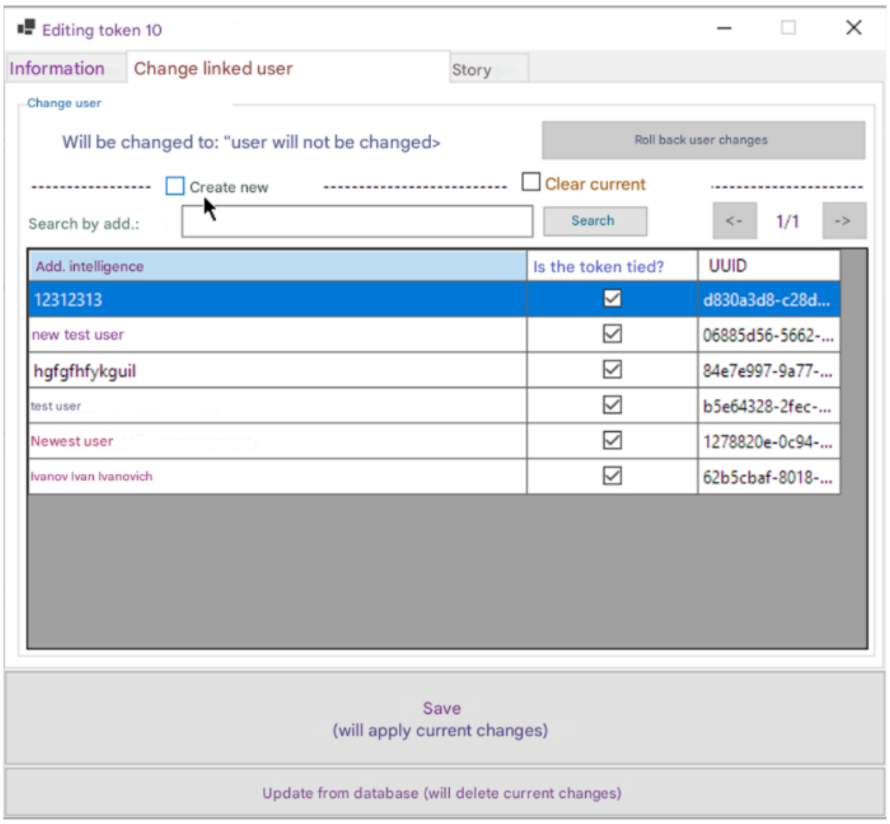
Изменить разрешённые IP

Сохранить
(применит текущие изменения)

Обновить из базы (удалит текущие изменения)

Picture 24 – Simultaneous editing of two tokens

The next tab of this window is for managing linked users.


Editing token 10

Information | Change linked user | Story

Change user

Will be changed to: "user will not be changed>

Roll back user changes

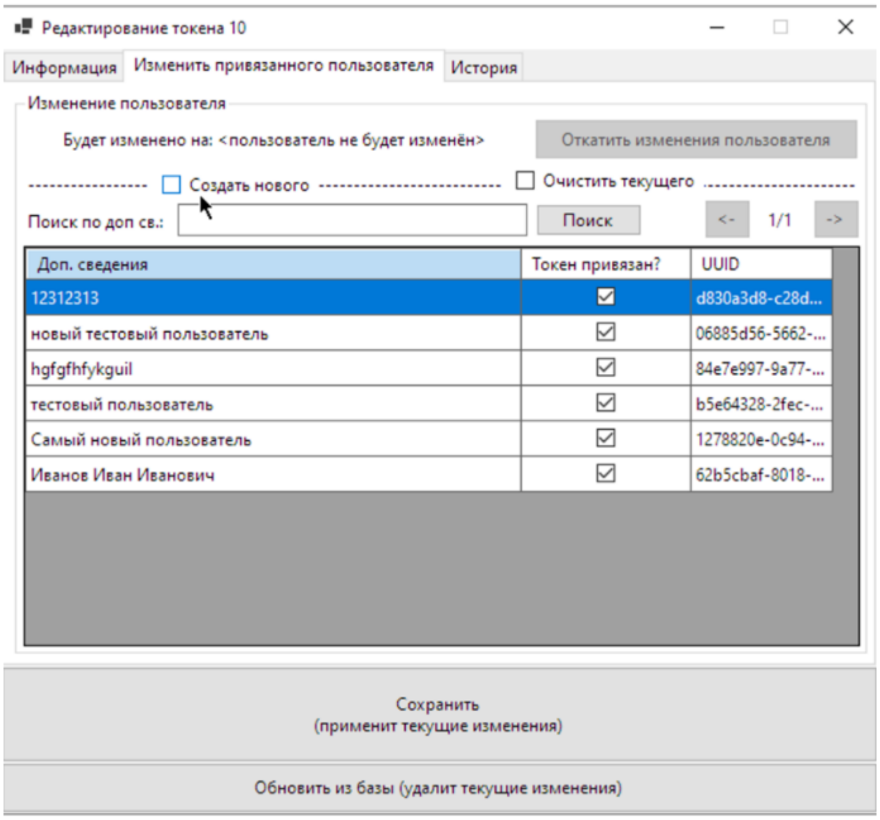
Create new | Clear current

Search by add.: | Search | <- 1/1 ->

Add. intelligence	Is the token tied?	UUID
12312313	☑	d830a3d8-c28d...
new test user	☑	06885d56-5662-...
hgfgfhfykguil	☑	84e7e997-9a77-...
test user	☑	b5e64328-2fec-...
Newest user	☑	1278820e-0c94-...
Ivanov Ivan Ivanovich	☑	62b5cbaf-8018-...

Save
(will apply current changes)

Update from database (will delete current changes)


Редактирование токена 10

Информация | Изменить привязанного пользователя | История

Изменение пользователя

Будет изменено на: <пользователь не будет изменён>

Откатить изменения пользователя

Создать нового | Очистить текущего

Поиск по доп св.: | Поиск | <- 1/1 ->

Доп. сведения	Токен привязан?	UUID
12312313	☑	d830a3d8-c28d...
новый тестовый пользователь	☑	06885d56-5662-...
hgfgfhfykguil	☑	84e7e997-9a77-...
тестовый пользователь	☑	b5e64328-2fec-...
Самый новый пользователь	☑	1278820e-0c94-...
Иванов Иван Иванович	☑	62b5cbaf-8018-...

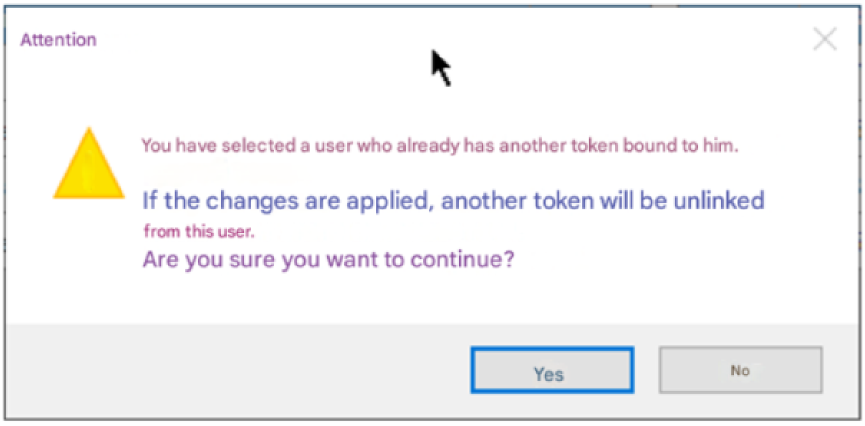
Сохранить
(применит текущие изменения)

Обновить из базы (удалит текущие изменения)

Picture 25 – Linked user management tab

Logically, this window can perform three main actions: delete a user, change a user, and create a new user.

Additionally, since only one user can be linked to a token, when attempting to add another user to which the token is already linked, the system will display a corresponding warning (this is shown below).

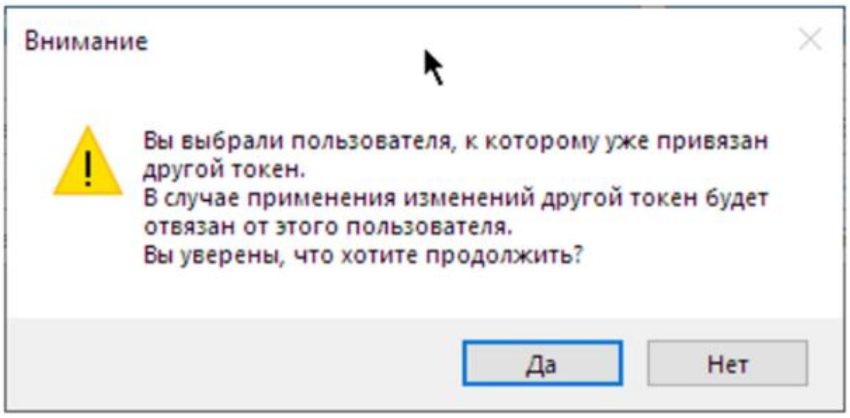


Attention

You have selected a user who already has another token bound to him.

If the changes are applied, another token will be unlinked from this user.

Are you sure you want to continue?

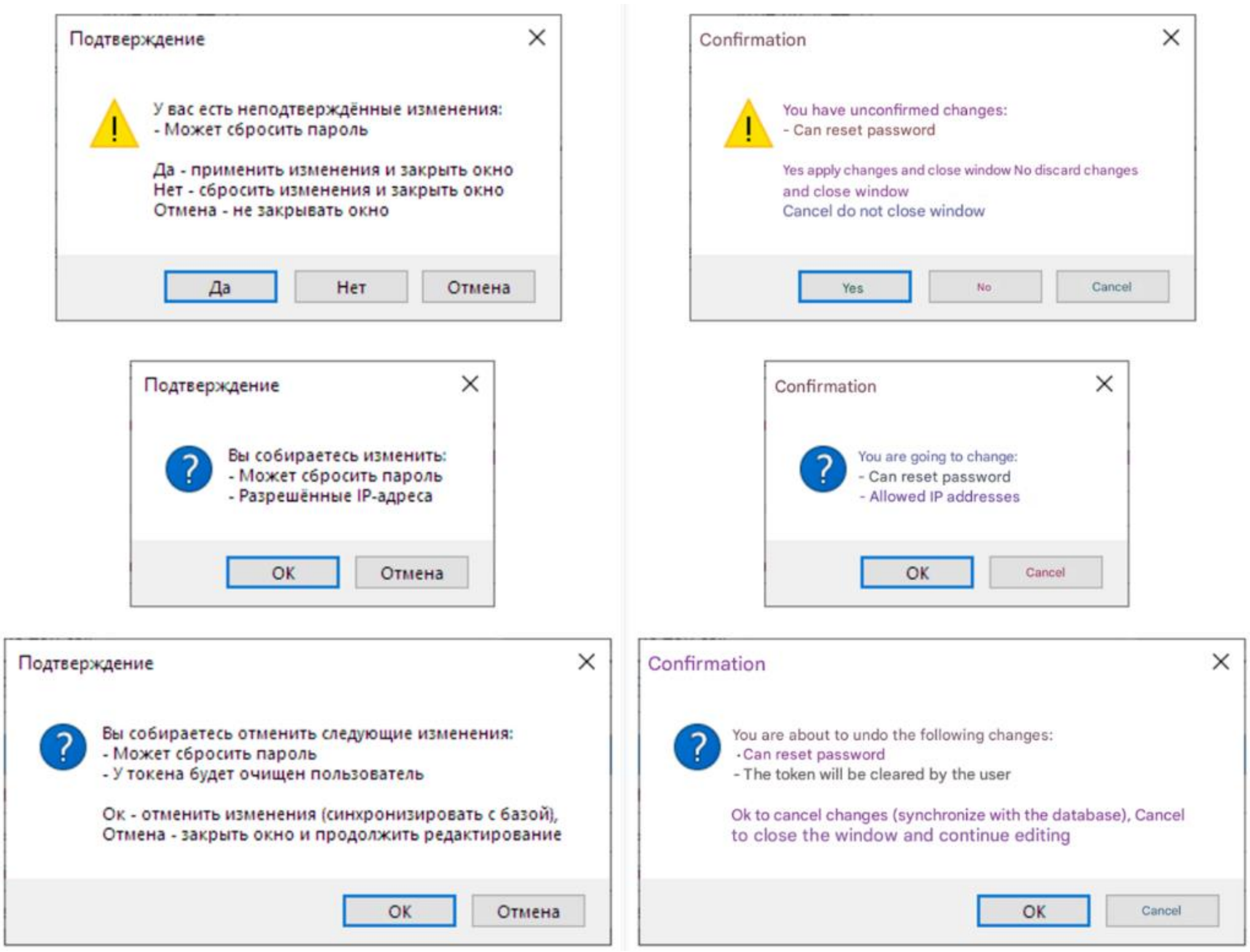


Внимание

Вы выбрали пользователя, к которому уже привязан другой токен.
В случае применения изменений другой токен будет отвязан от этого пользователя.
Вы уверены, что хотите продолжить?

Picture 26 – User unlinking warning

The approach to applying and canceling changes made to the token deserves special attention. When attempting to save or roll back all changes, as well as when closing the window, the system will first display a warning listing all changes that have been made but have not yet applied at the database level.


Подтверждение

У вас есть неподтверждённые изменения:
- Может сбросить пароль

Да - применить изменения и закрыть окно
Нет - сбросить изменения и закрыть окно
Отмена - не закрывать окно

Confirmation

You have unconfirmed changes:
- Can reset password

Yes apply changes and close window No discard changes and close window
Cancel do not close window

Подтверждение

Вы собираетесь изменить:
- Может сбросить пароль
- Разрешённые IP-адреса

Confirmation

You are going to change:
- Can reset password
- Allowed IP addresses

Подтверждение

Вы собираетесь отменить следующие изменения:
- Может сбросить пароль
- У токена будет очищен пользователь

Ок - отменить изменения (синхронизировать с базой),
Отмена - закрыть окно и продолжить редактирование

Confirmation

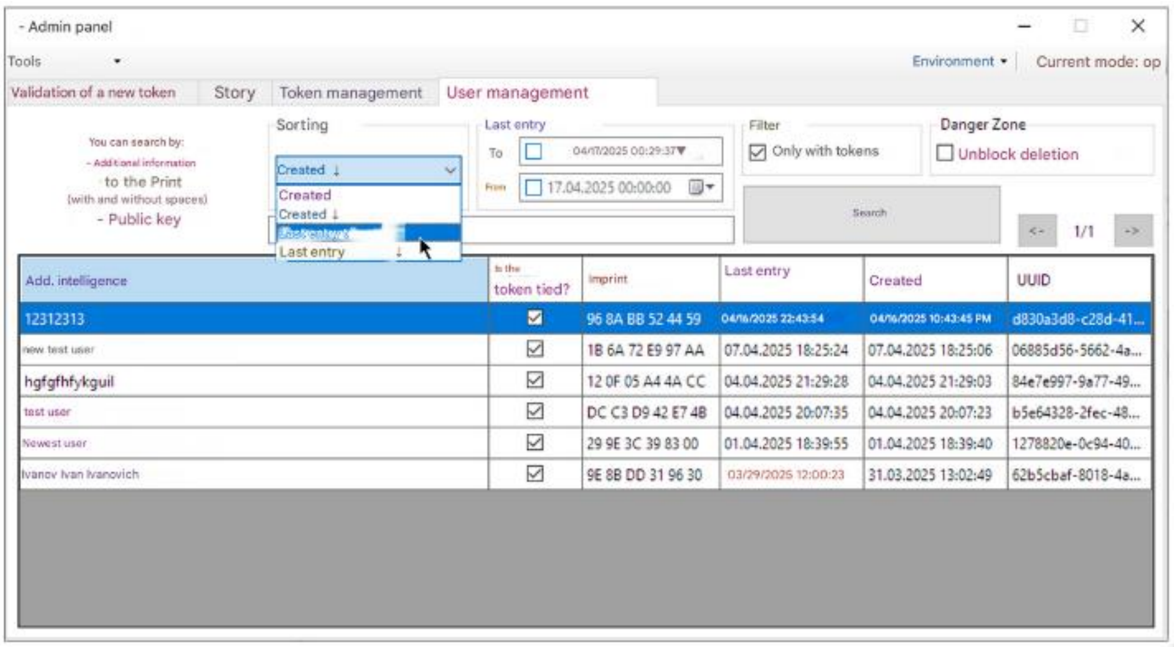
You are about to undo the following changes:
- Can reset password
- The token will be cleared by the user

Ok to cancel changes (synchronize with the database), Cancel to close the window and continue editing

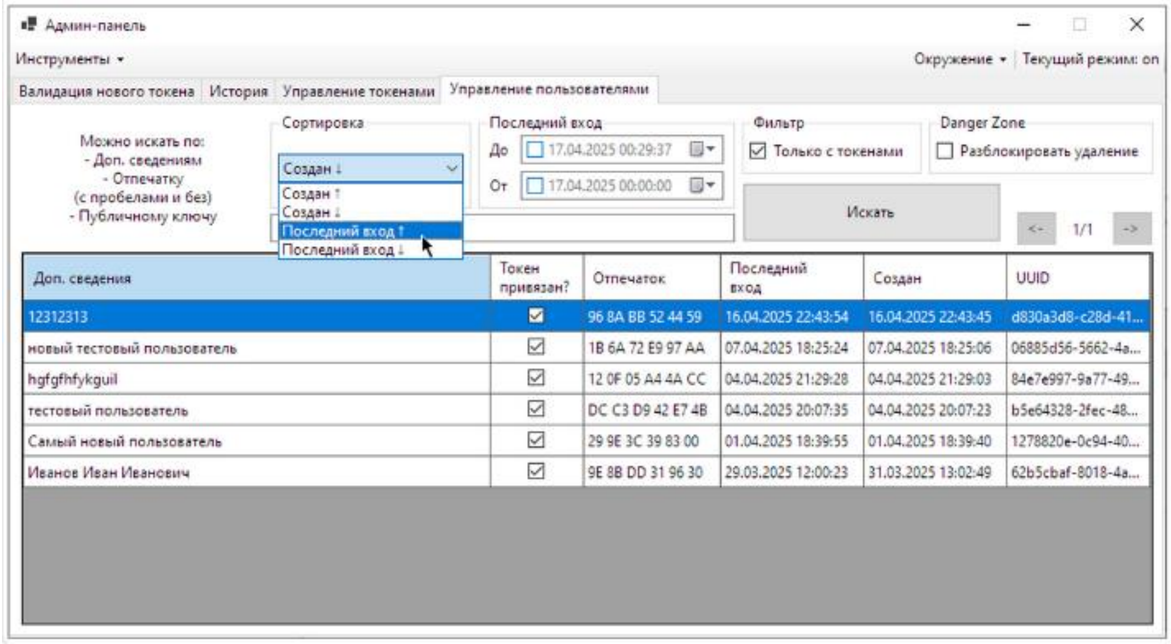
Picture 27 – Confirmation windows when editing a token

2.3.2.4 User management window

The interface of the user management window is shown below.



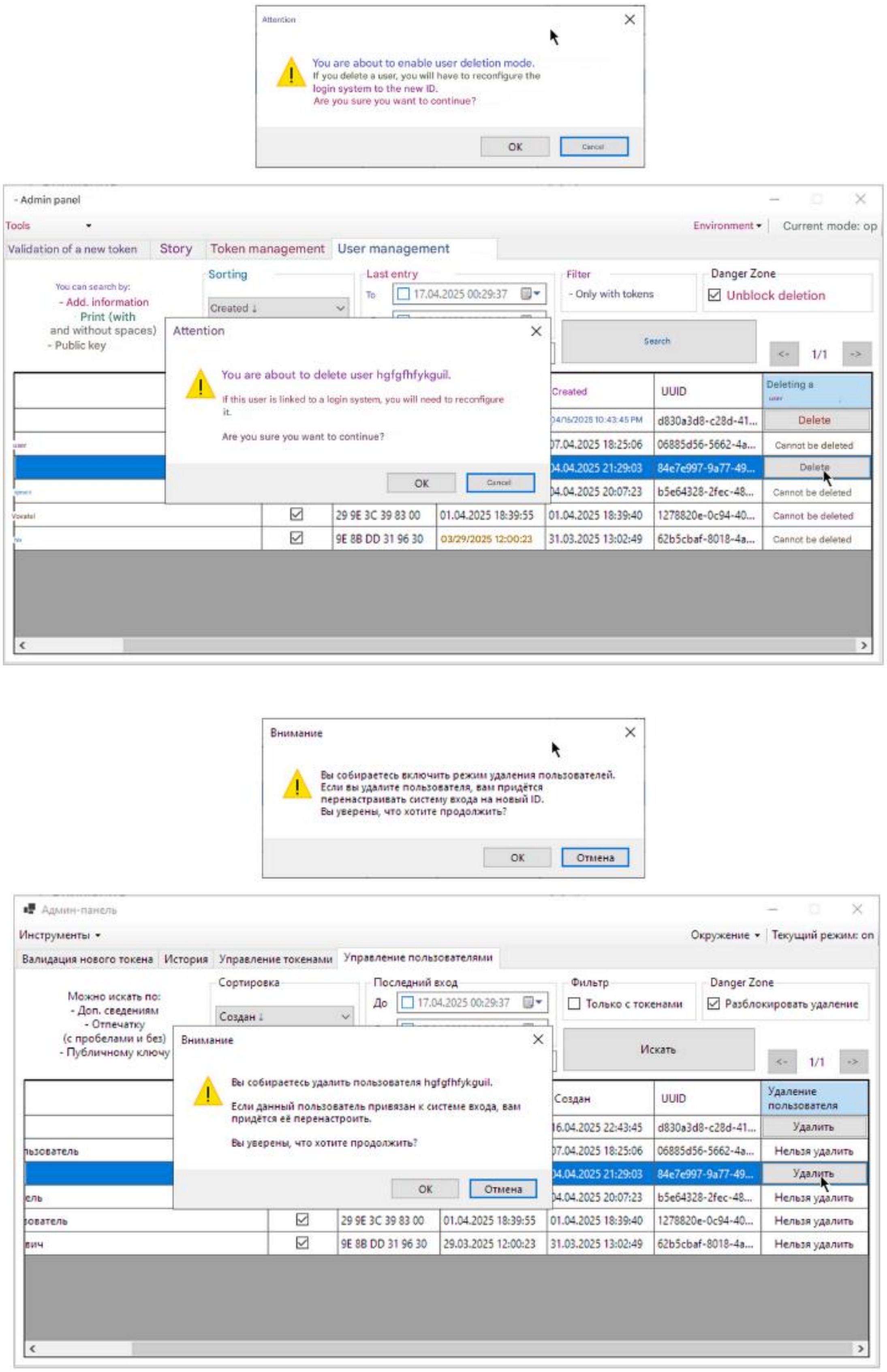
Add. intelligence	Is the token tied?	Imprint	Last entry	Created	UUID
12312313	☑	96 8A BB 52 44 59	04/16/2025 22:43:54	04/16/2025 10:43:45 PM	d830a3d8-c28d-41...
new test user	☑	1B 6A 72 E9 97 AA	07.04.2025 18:25:24	07.04.2025 18:25:06	06885d56-5662-4a...
hgfgfhfykguil	☑	12 0F 05 A4 4A CC	04.04.2025 21:29:28	04.04.2025 21:29:03	84e7e997-9a77-49...
test user	☑	DC C3 D9 42 E7 4B	04.04.2025 20:07:35	04.04.2025 20:07:23	b5e64328-2fec-48...
Newest user	☑	29 9E 3C 39 83 00	01.04.2025 18:39:55	01.04.2025 18:39:40	1278820e-0c94-40...
Ivanov Ivan Ivanovich	☑	9E 8B DD 31 96 30	03/29/2025 12:00:23	31.03.2025 13:02:49	62b5cbaf-8018-4a...



Доп. сведения	Токен привязан?	Отпечаток	Последний вход	Создан	UUID
12312313	☑	96 8A BB 52 44 59	16.04.2025 22:43:54	16.04.2025 22:43:45	d830a3d8-c28d-41...
новый тестовый пользователь	☑	1B 6A 72 E9 97 AA	07.04.2025 18:25:24	07.04.2025 18:25:06	06885d56-5662-4a...
hgfgfhfykguil	☑	12 0F 05 A4 4A CC	04.04.2025 21:29:28	04.04.2025 21:29:03	84e7e997-9a77-49...
тестовый пользователь	☑	DC C3 D9 42 E7 4B	04.04.2025 20:07:35	04.04.2025 20:07:23	b5e64328-2fec-48...
Самый новый пользователь	☑	29 9E 3C 39 83 00	01.04.2025 18:39:55	01.04.2025 18:39:40	1278820e-0c94-40...
Иванов Иван Иванович	☑	9E 8B DD 31 96 30	29.03.2025 12:00:23	31.03.2025 13:02:49	62b5cbaf-8018-4a...

Picture 28 – User management window

The approach to deleting users deserves attention. First, deletion must be unlocked separately by clicking the checkbox five times, after which a new table row will be shown with buttons allowing you to delete users who are not linked to tokens (this is shown below).



Attention

You are about to enable user deletion mode.
If you delete a user, you will have to reconfigure the login system to the new ID.
Are you sure you want to continue?

OK Cancel

- Admin panel

Tools ▾ Environment ▾ Current mode: op

Validation of a new token | Story | Token management | User management

You can search by:
- Add. information
Print (with and without spaces)
- Public key

Sorting: Created ↓

Last entry: To 17.04.2025 00:29:37

Filter: - Only with tokens

Danger Zone: ☑ Unblock deletion

Search <- 1/1 ->

Attention

You are about to delete user hgfgfhfykguil.

If this user is linked to a login system, you will need to reconfigure it.

Are you sure you want to continue?

OK Cancel

				Created	UUID	Deleting a user
				04/16/2025 10:43:45 PM	d830a3d8-c28d-41...	Delete
user				07.04.2025 18:25:06	06885d56-5662-4a...	Cannot be deleted
				04.04.2025 21:29:03	84e7e997-9a77-49...	Delete
				04.04.2025 20:07:23	b5e64328-2fec-48...	Cannot be deleted
Vovatel	☑	29 9E 3C 39 83 00	01.04.2025 18:39:55	01.04.2025 18:39:40	1278820e-0c94-40...	Cannot be deleted
	☑	9E 8B DD 31 96 30	03/29/2025 12:00:23	31.03.2025 13:02:49	62b5cbaf-8018-4a...	Cannot be deleted

Внимание

Вы собираетесь включить режим удаления пользователей.
Если вы удалите пользователя, вам придётся перенастраивать систему входа на новый ID.
Вы уверены, что хотите продолжить?

OK Отмена

Админ-панель

Инструменты ▾ Окружение ▾ Текущий режим: оп

Валидация нового токена | История | Управление токенами | Управление пользователями

Можно искать по:
- Доп. сведениям
- Отпечатку
(с пробелами и без)
- Публичному ключу

Сортировка: Создан ↓

Последний вход: До 17.04.2025 00:29:37

Фильтр: ☐ Только с токенами

Danger Zone: ☑ Разблокировать удаление

Искать <- 1/1 ->

Внимание

Вы собираетесь удалить пользователя hgfgfhfykguil.

Если данный пользователь привязан к системе входа, вам придётся её перенастроить.

Вы уверены, что хотите продолжить?

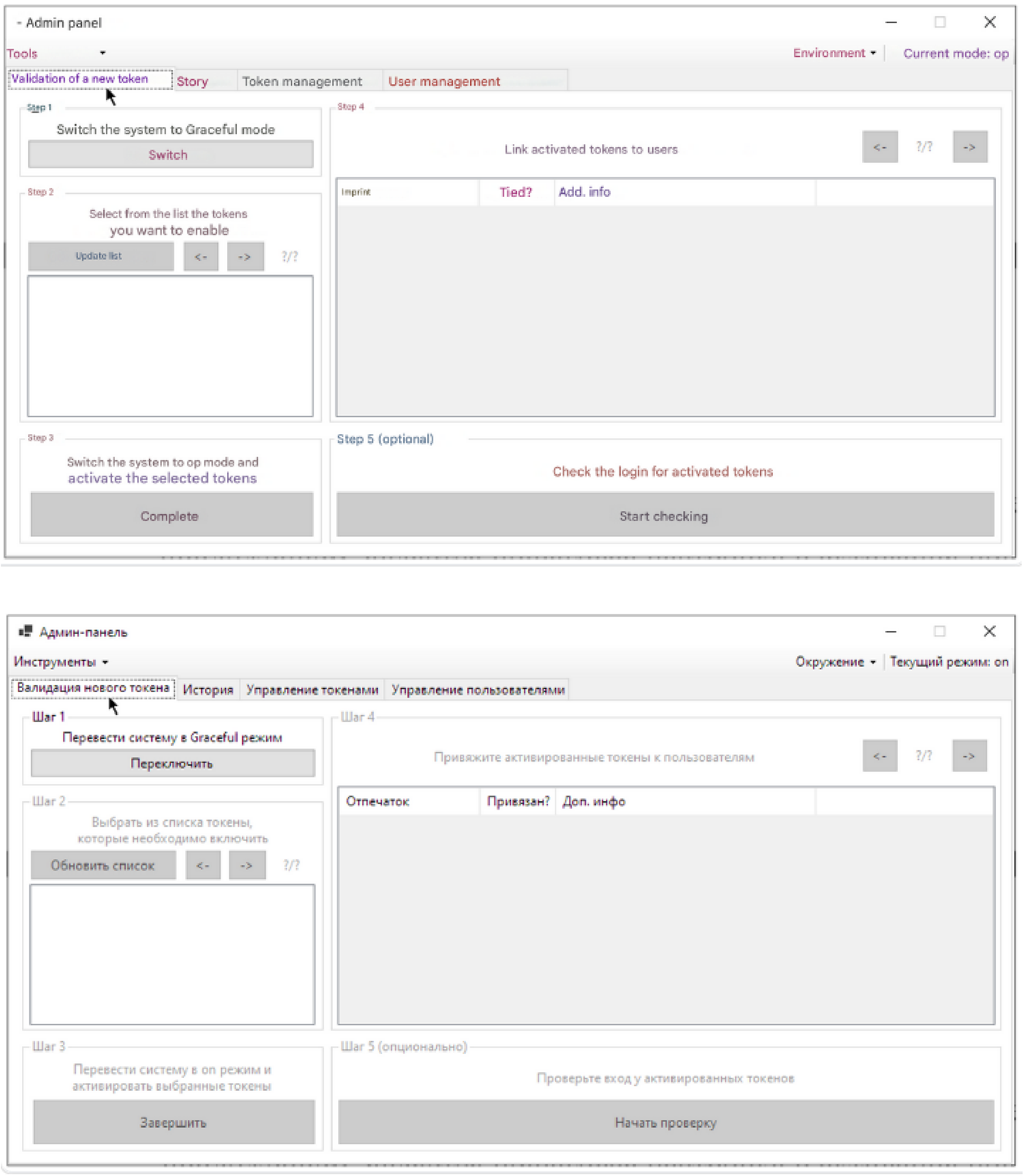
OK Отмена

				Создан	UUID	Удаление пользователя
				16.04.2025 22:43:45	d830a3d8-c28d-41...	Удалить
льзователь				07.04.2025 18:25:06	06885d56-5662-4a...	Нельзя удалить
				04.04.2025 21:29:03	84e7e997-9a77-49...	Удалить
ель				04.04.2025 20:07:23	b5e64328-2fec-48...	Нельзя удалить
зователь	☑	29 9E 3C 39 83 00	01.04.2025 18:39:55	01.04.2025 18:39:40	1278820e-0c94-40...	Нельзя удалить
вич	☑	9E 8B DD 31 96 30	29.03.2025 12:00:23	31.03.2025 13:02:49	62b5cbaf-8018-4a...	Нельзя удалить

Picture 29 – User deletion

2.3.2.5 New token configuration window

The last use case provided by the admin-panel is the ability to configure and activate a new token step by step. The layout of this window is shown below.


Admin panel
Tools
Environment
Current mode: op
Validation of a new token
Story
Token management
User management
Step 1
Switch the system to Graceful mode
Switch
Step 2
Select from the list the tokens you want to enable
Update list
<-
->
?/?
Step 3
Switch the system to op mode and activate the selected tokens
Complete
Step 4
Link activated tokens to users
Imprint
Tied?
Add. info
Step 5 (optional)
Check the login for activated tokens
Start checking
Админ-панель
Инструменты
Окружение
Текущий режим: on
Валидация нового токена
История
Управление токенами
Управление пользователями
Шаг 1
Перевести систему в Graceful режим
Переключить
Шаг 2
Выбрать из списка токены, которые необходимо включить
Обновить список
Шаг 3
Перевести систему в on режим и активировать выбранные токены
Завершить
Шаг 4
Привяжите активированные токены к пользователям
Отпечаток
Привязан?
Доп. инфо
Шаг 5 (опционально)
Проверьте вход у активированных токенов
Начать проверку

Picture 30 – New token validation window

Although the steps are numbered sequentially, the two columns are independent of each other (the result of the first column is displayed cumulatively in the second column).

The requirement for such a system is due to an additional security measure presence: tokens are initially created inactive and unlinked. Therefore, an attacker would not only have to add their token to the system but also edit it in an illegitimate manner.

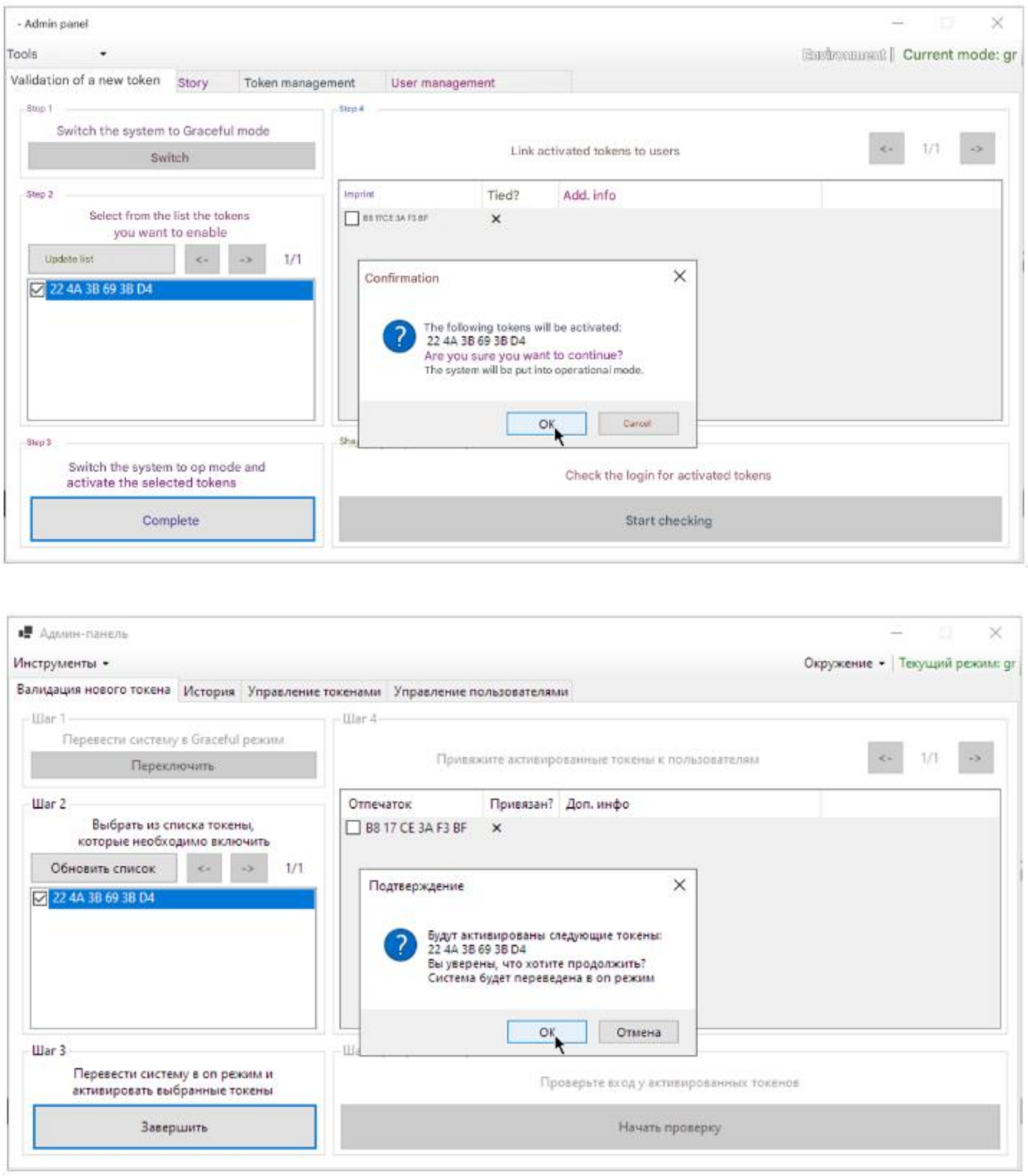
Thus, for a new token to enter the system, it is necessary to:

- switch the system to a “graceful” mode (allow the registration of new tokens),
- mark new token as “active”,
- return the system to normal operation mode.

The first three steps are responsible for this.

Then, for the new tokens to work properly, they need to be linked to users. The second column is responsible for this.

The following scenario is shown below: one token has already been activated and has reached step 4, another is at step 2, and the user is about to complete step 3.



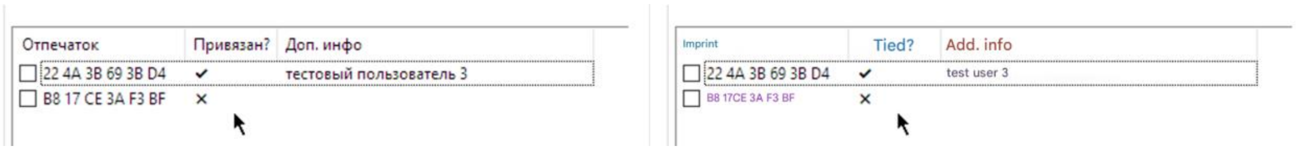
- Admin panel
Tools
Environment | Current mode: gr
Validation of a new token
Story
Token management
User management
Step 1
Switch the system to Graceful mode
Switch
Step 2
Select from the list the tokens you want to enable
Update list
<-
->
1/1
22 4A 3B 69 3B D4
Step 3
Switch the system to op mode and activate the selected tokens
Complete
Step 4
Link activated tokens to users
Imprint
Tied?
Add. info
B8 17CE 3A F3 BF
Check the login for activated tokens
Start checking
Confirmation
OK
Cancel
Админ-панель
Инструменты
Окружение | Текущий режим: gr
Валидация нового токена
История
Управление токенами
Управление пользователями
Шаг 1
Перевести систему в Graceful режим
Переключить
Шаг 2
Выбрать из списка токены, которые необходимо включить
Обновить список
Шаг 3
Перевести систему в on режим и активировать выбранные токены
Завершить
Шаг 4
Привяжите активированные токены к пользователям
Отпечаток
Привязан?
Доп. инфо
B8 17 CE 3A F3 BF
Проверьте вход у активированных токенов
Начать проверку
Подтверждение
Отмена

The following tokens will be activated:
22 4A 3B 69 3B D4
Are you sure you want to continue?
The system will be put into operational mode.

Будут активированы следующие токены:
22 4A 3B 69 3B D4
Вы уверены, что хотите продолжить?
Система будет переведена в on режим

Picture 31 – Completing stage 3 of token activation

After that, the system administrator is prompted to bind new tokens to users. The result of this linking is displayed visually in a separate column in the list.

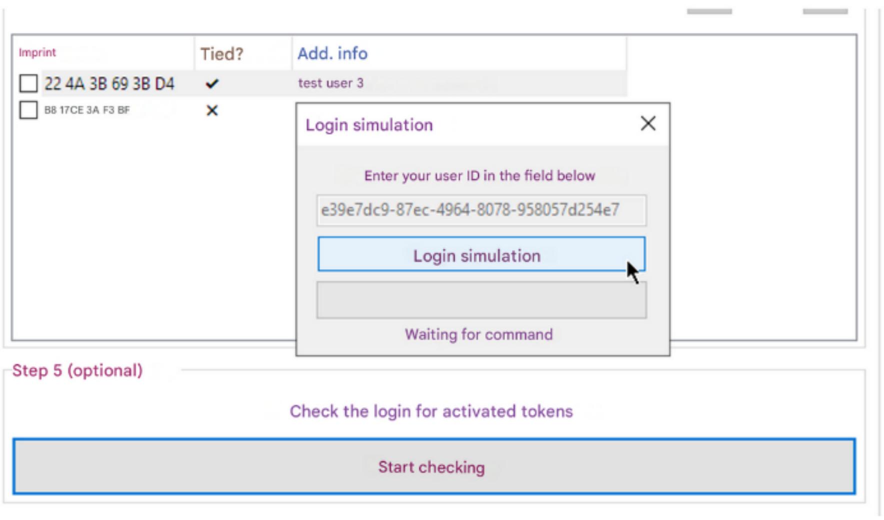


Отпечаток	Привязан?	Доп. инфо
☐ 22 4A 3B 69 3B D4	✓	тестовый пользователь 3
☐ B8 17 CE 3A F3 BF	×	

Imprint	Tied?	Add. info
☐ 22 4A 3B 69 3B D4	✓	test user 3
☐ B8 17CE 3A F3 BF	×	

Picture 32 – User binding result

As a final step, the administrator can test the new tokens through a login simulation. At this stage, login windows are opened sequentially for each linked but not yet verified user.



Imprint	Tied?	Add. info
☐ 22 4A 3B 69 3B D4	✓	test user 3
☐ B8 17CE 3A F3 BF	✕	

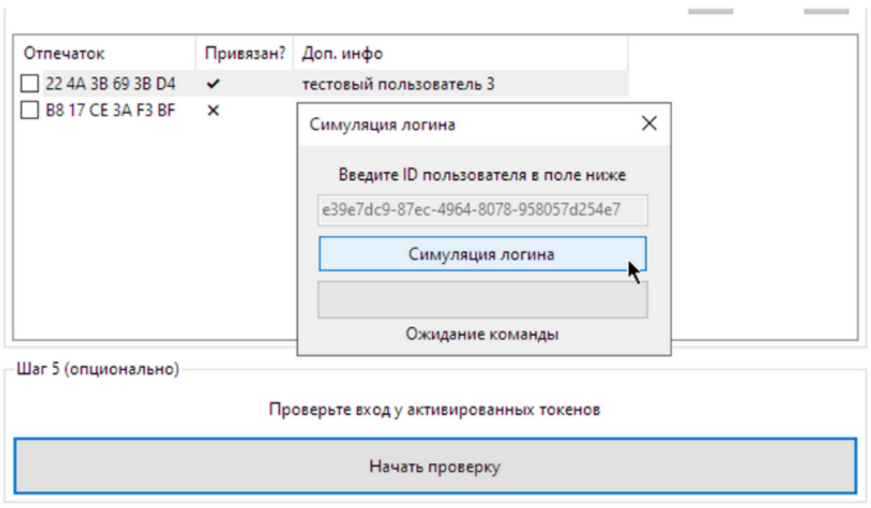
Enter your user ID in the field below

e39e7dc9-87ec-4964-8078-958057d254e7

Waiting for command

Step 5 (optional)

Check the login for activated tokens



Отпечаток	Привязан?	Доп. инфо
☐ 22 4A 3B 69 3B D4	✓	тестовый пользователь 3
☐ B8 17 CE 3A F3 BF	✕	

Введите ID пользователя в поле ниже

e39e7dc9-87ec-4964-8078-958057d254e7

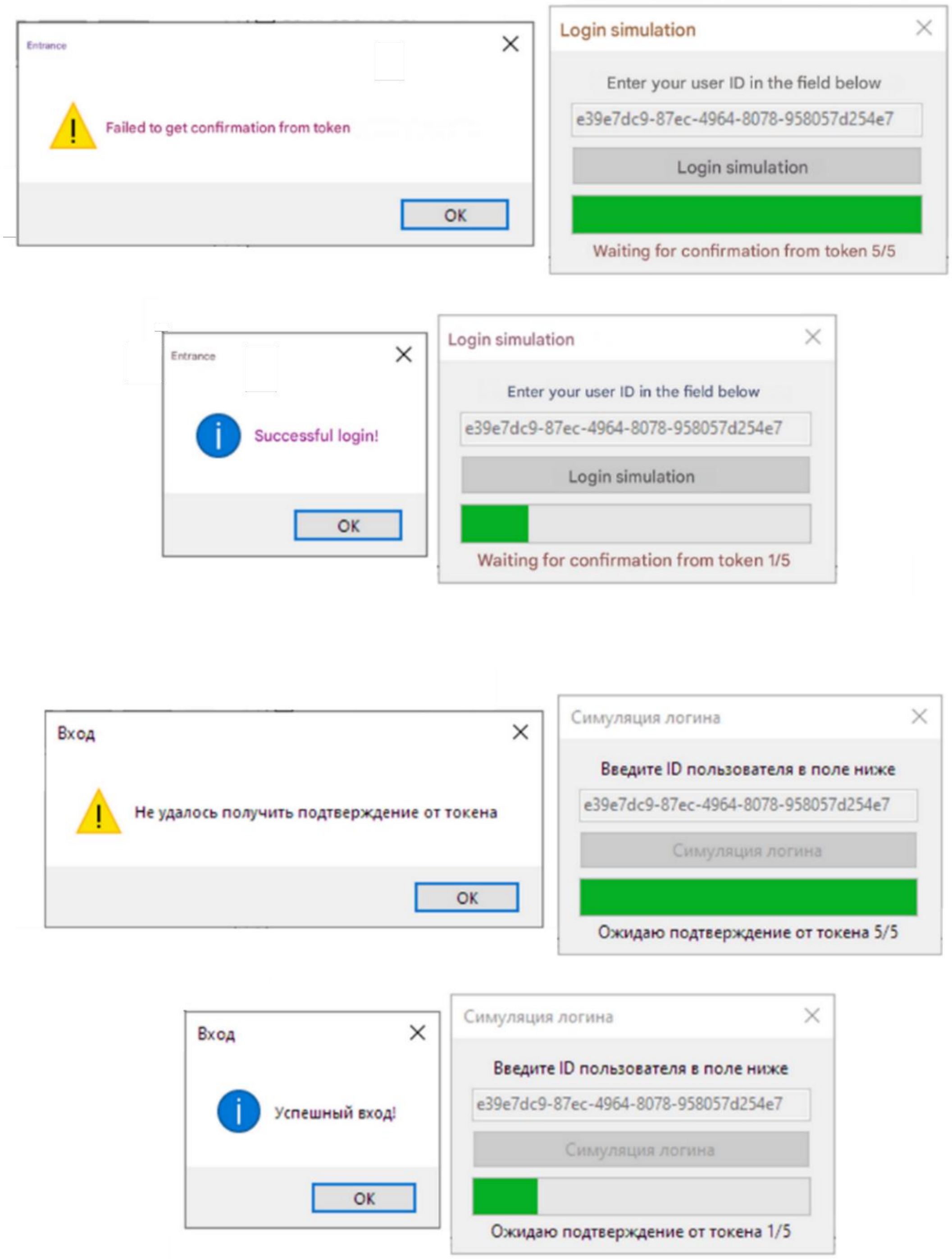
Ожидание команды

Шаг 5 (опционально)

Проверьте вход у активированных токенов

Picture 33 – Login simulation for new tokens

The login simulation window starts with a timer that polls the server for login availability. At the end, it either displays a success message and closes, or displays an error and waits for a repeat command.


Entrance

Failed to get confirmation from token

OK

Login simulation

Enter your user ID in the field below

e39e7dc9-87ec-4964-8078-958057d254e7

Login simulation

Waiting for confirmation from token 5/5

Entrance

Successful login!

OK

Login simulation

Enter your user ID in the field below

e39e7dc9-87ec-4964-8078-958057d254e7

Login simulation

Waiting for confirmation from token 1/5

Вход

Не удалось получить подтверждение от токена

OK

Симуляция логина

Введите ID пользователя в поле ниже

e39e7dc9-87ec-4964-8078-958057d254e7

Симуляция логина

Ожидаю подтверждение от токена 5/5

Вход

Успешный вход!

OK

Симуляция логина

Введите ID пользователя в поле ниже

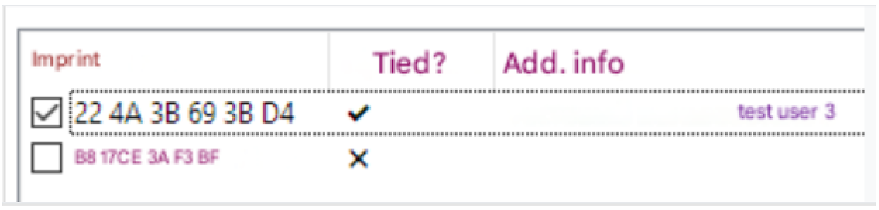
e39e7dc9-87ec-4964-8078-958057d254e7

Симуляция логина

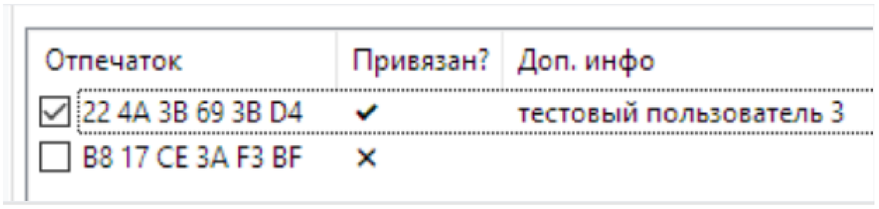
Ожидаю подтверждение от токена 1/5

Picture 34 – Successful and unsuccessful login simulation

After that, verified users are marked with an additional check mark in the interface.



Imprint	Tied?	Add. info
☑ 22 4A 3B 69 3B D4	✔	test user 3
☐ B8 17CE 3A F3 BF	✕	

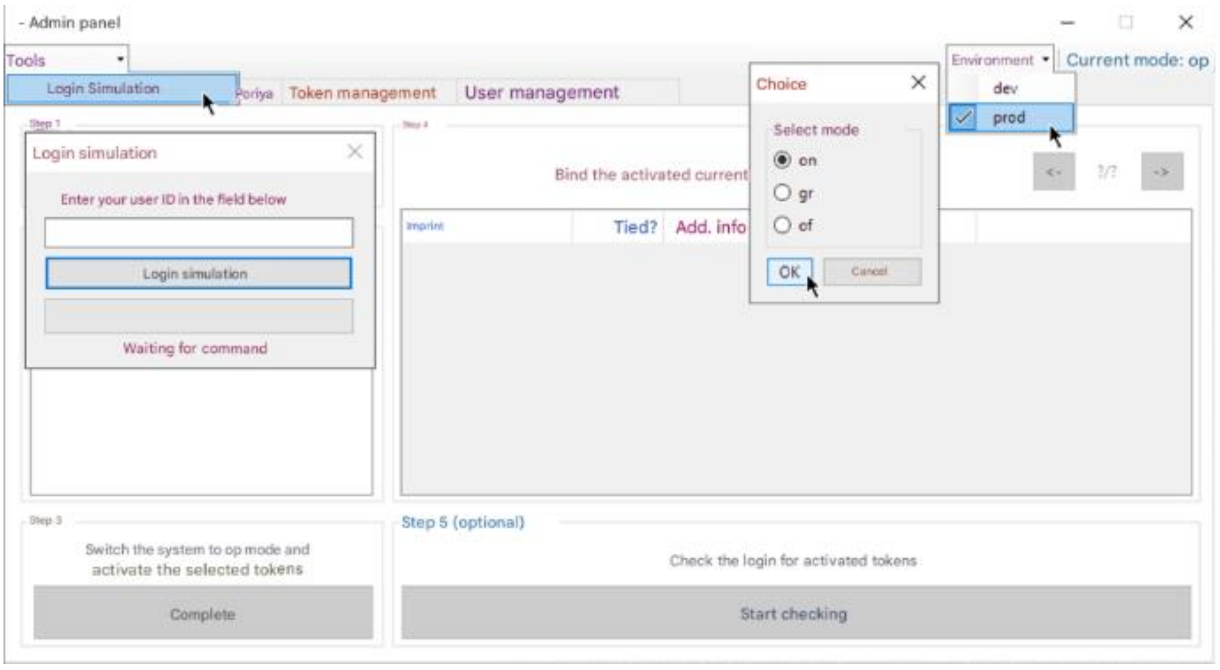


Отпечаток	Привязан?	Доп. инфо
☑ 22 4A 3B 69 3B D4	✔	тестовый пользователь 3
☐ B8 17 CE 3A F3 BF	✕	

Picture 35 – Verified users marking

2.3.2.6 Changing application settings and additional functions

The combined screenshot below shows all the settings and additional features of the admin panel.



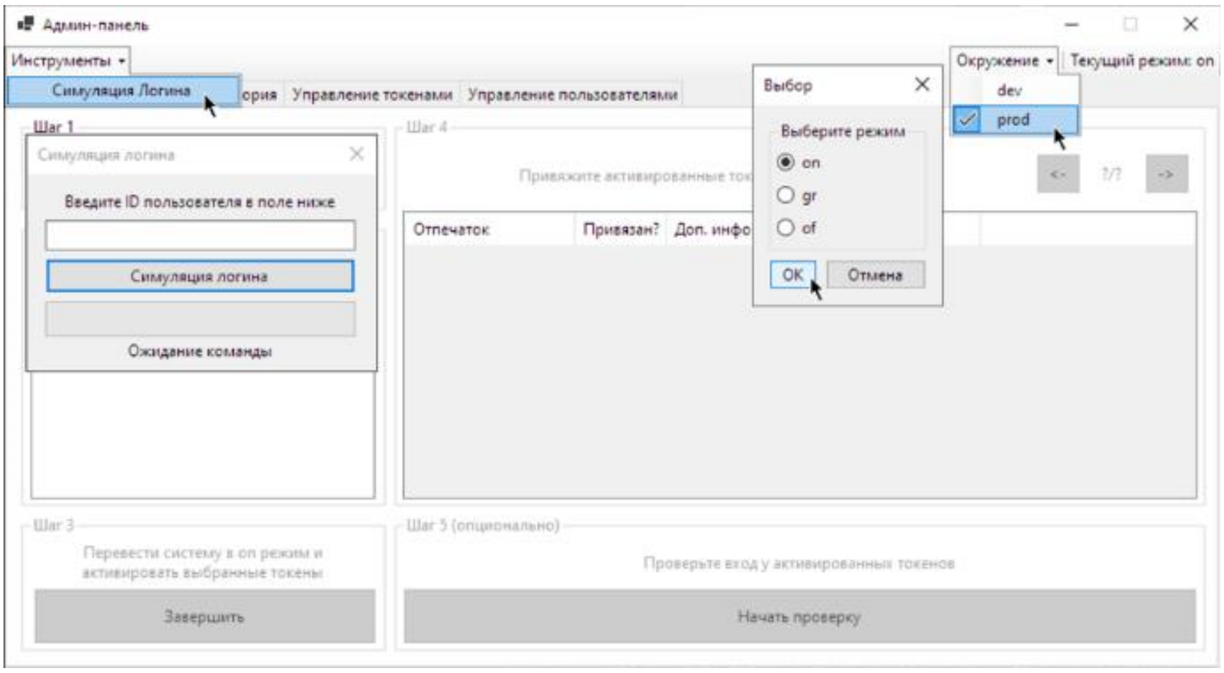
Enter your user ID in the field below

Waiting for command

Bind the activated current

Switch the system to op mode and activate the selected tokens

Check the login for activated tokens



Введите ID пользователя в поле ниже

Ожидание команды

Привяжите активированные ток

Перевести систему в on режим и активировать выбранные токены

Проверьте вход у активированных токенов

Picture 36 – Additional settings

These include:

– freely launching login simulation,

– changing the system's operating mode (you can also set the “of” mode, which will disable all checks specified in the “validation” block),

– changing the environment.

2.4 DevOps and CI/CD

During server development automatic delivery and deployment mechanisms were used to reduce downtime.

Additionally, in real-world operation, this can serve as a security mechanism: developers may not be allowed to have direct access to the client's servers, instead relying on automated delivery mechanisms.

2.4.1 Containerization

Docker was used for the application containerization, Dockerfile performs the following actions:

- Copying application files into a container,
- installing Python libraries,
- applying Django migrations,
- running the setup_app command (it fills in the gaps in the AppSettingsModel with the keys and default settings),
- running gunicorn HTTP-server on port 8000.

Docker compose was used to automatically apply container settings (application uses only one container labeled “diplomabackend”):

- port 8000 is opened to the local system.
- a symbolic link for the database file is created.
- Two environment variables are set: the one is settings set used by the application (could be dev or prod, influences HTTPS enforcement and debug mode); the other is IPs allowed for administrative requests (limits admin-panel access only for certain Ips and subnets).

2.4.2 CI/CD

Two git branches were used during the development: dev and main.

A GitHub Actions workflow is automatically triggered each time a commit is made to the main branch. It performs the following operations:

- shuts down any running containers with the application,
- pulls most recent changes from the origin/main,
- rebuilds the container and launches an updated version.
- performs checks on a running container:

1) it started without errors,
2) the server is running in production mode,
3) the server has passed a health check,

If the checks fail, the container is prematurely terminated to avoid information leaks and data loss.

3 Information security analysis

3.1 Analysis of potential vulnerabilities

This section discusses analysis of potential vulnerabilities typical of two-factor authentication systems, which I used as a basis for developing an application architecture that eliminates them.

3.1.1 Communication channel

Since all communication is conducted via the Internet, the following scenarios are possible:

- traffic sniffing (would allow the attacker to determine the fact of token activation; this is the only vulnerability that was accepted);
- reading the communication between server and token.

To prevent the latter vulnerability, application uses HTTPS, which is managed by the Nginx web server functioning in reverse proxy mode.

Additionally, to further prevent man-in-the-middle attacks, system administrators are supposed to manually verify token imprints during token registration.

3.1.2 Token

Two types of confidentiality breaches are possible:

- creating a full copy of the device's internal memory,
- tracking user actions through spyware or through observation.

Additionally, obtaining unauthorized physical access to the device is also possible.

3.1.3 Additional requirements

Even when the application is used legitimately, some potential vulnerabilities are possible:

- use of an outdated (and thus less secure) application version,

– presence of old but still active tokens.

Furthermore, since token login confirmation is valid for a certain time frame, an attacker can potentially try to authenticate at the same time (double login vulnerability).

Finally, the system must be resistant to fake tokens creation (only allow new tokens when necessary).

3.2 Vulnerability mitigation

Below is a summary table showing the methods used in the application to eliminate vulnerabilities listed above.

Table 1 – Vulnerabilities and Requirements Addressed

Vulnerability/Requirement	**How it was eliminated**
Compromise of the communication channel	TLS
Traffic identification	—
Logical access to the token	Hardware key storage
Physical access to token	Biometrics, PIN-code
Moving a token outside the boundaries of the organization	IP based restrictions
Logical access to server administration	Prohibition of access outside the controlled area
Protection against double login	A separate mechanism
System integrity (system is a product)	Absence of external actors
Flooding protection	System operating modes
Additional requirements	Versioning and automatic token expiration
Ease of use of the application	Desing optimizations
User self-diagnostic	Last logins screen
Diagnostics by the administrator	Technical and service data output
Supporting the functional requirements of the system administrator	Various admin panel modes: monitoring, technical support, step-by-step configuration
Development optimization	DevOps and CI/CD

LINKS

All files (text and diagrams) in Russian are available on Google Drive: https://drive.google.com/drive/folders/1gXfIuAmHMhwrkAZPj5cLDALVC4FxuiQX

Translated versions can be accessed via the following link: https://drive.google.com/drive/folders/1A4dDpy3mjZwy8Cetw006S4YqfEzXqQju

The source code for all the parts of the software package is available on my personal GitHub:

– backend: https://github.com/Zaqzxcswsde/diplomabackend;
– mobile app: https://github.com/Zaqzxcswsde/diplomamobile;
– admin-panel: https://github.com/Zaqzxcswsde/diplomaadminpanel.

The production server can be accessed via the following URL: https://zaqzxcswsde.ru.

API specification (without comments) can be found through this link.