UNLIMITED PRODUCTION

eXpress Communication System

Microsoft Outlook Add-in

Installation Guide

ADFS Version 3.1.0.2

Mailing address:	127030, Moscow, 24/1 Novoslobodskaya Street,
Phone:	+7 (499) 288-01-22
E-mail:	sales@express.ms
Web:	https://express.ms/

TABLE OF CONTENTS

TERMS AND DEFINITIONS

Term	Definition
AD	Active Directory — Microsoft's directory service for the Windows Server operating system family
ADFS	Active Directory Federation Services — a software component developed by Microsoft that provides users with Single Sign-On (SSO) capabilities to systems and applications located outside organizational boundaries
API	Application Programming Interface — an interface for interaction between programs and applications
Chat Bot	Conference Notifier Bot chat bot
Conference Creator	The user who creates and can edit an Express conference using the Outlook add-in
Container	A standardized, isolated, and portable software package that includes everything needed to run an application, including code, runtime, system tools, libraries, and settings.
CTS	Corporate Transport Server
DB	Database
DBMS	Database Management System
Docker	An open-source containerization platform that automates application creation, delivery, and management
Docker-compose	An add-on (utility) over Docker, a Python application that allows running multiple containers simultaneously and routing data flows between them.
ETS	Enterprise Transport Server
express-core-service	A server application for the Outlook add-in that provides core functionality (creating and modifying conferences)
express-template-service	A server application for the Outlook add-in that provides additional functionality (creating and modifying email templates, text, logo, signature, etc.)
Express CS, Express, System	Express Communication System
Invited Participant	User(s) invited by the Express conference creator using the Outlook add-in
JSON	A text-based data exchange format based on JavaScript
Log	An entry in the server and/or client event log
MS Exchange	Microsoft Mail Server
MS Outlook	Microsoft mail client, part of the MS Office application package
OS	Operating System
PC	Personal Computer
Single CTS	Unified Corporate Transport Server
Split CTS	Split Corporate Transport Server: Front CTS and Back CTS
SW	Software

Chapter 1

GENERAL REQUIREMENTS

Attention! The following competencies are required to perform the operations in this instruction:

- Windows Server administration;
- AFDS administration;
- PostgreSQL administration;
- Linux administration;
- Docker administration;
- eXpress administration;
- understanding of JSON.

Before starting, ensure that software and hardware meet the following requirements:

- Docker version not lower than Docker version 24.0.7;
- Docker-compose version not lower than version 1.29.2;
- Database requirements:
    - PostgreSQL version not lower than (Debian 13.11-0+deb11u1, Windows);
    - superuser privileges on the PostgreSQL server;
    - a dedicated PostgreSQL database and a dedicated PostgreSQL database user: this user must be the owner of this database and have all privileges on it;
- ADFS solution installed and configured, version 22.0.1 or higher;
- Outlook client version 2013 or higher;
- Outlook must be used with a mailbox that corresponds to the e-mail address registered in the Express CS account;
- Express CS solution installed and configured, version 3.8 or higher;
- The Conference Notifier Bot must be configured in Express CS. Requirements for configuring the chat bot and methods for verifying the correctness of the configuration are described in Chapter 3 "Installing and Configuring Conference Notifier Chat Bot".
- Network interaction between components must be configured using the corresponding ports;
- NET Framework 4.8 and the latest Windows updates for WebView2 Runtime version 141.0.3537.57 or higher must be installed on client PCs (https://developer.microsoft.com/ru-ru/microsoft-edge/webview2/?form=MA13LH).

If there are no other services on the virtual machine, the Docker server must meet the following technical requirements (Table 1):

Table 1

Component	**Parameters**
Processor	64-bit 2.4 GHz processor 2 core
RAM	4 GB
Operating System	Debian GNU/Linux 11 or later
Hard Disk	40 GB or more

The PostgreSQL DBMS can be located on a dedicated server running Windows OS or Linux OS. If a pre-installed PostgreSQL DBMS is absent, and the deployed solution architecture includes the Express CS Bot server, then the PostgreSQL docker container, which is part of the Bot server, can be used to create the DB. A Linux server on which the Outlook add-in will be deployed can be used to create the PostgreSQL DB using the PostgreSQL docker container

Attention! It is forbidden to use the following Express CS servers to create the PostgreSQL DB: CTS Back, Front, Media, Rec, ETS, and others (except for the Bot server).

If there are no other services on the virtual machine, the PostgreSQL DBMS server running Linux OS must meet the following technical requirements (Table 2):

Table 2

Component	**Parameters**
Processor	64-bit 2.4 GHz processor 2 core
RAM	4 GB
Operating System	Debian GNU/Linux 11 or later
Hard Disk	40 GB or more

If the PostgreSQL DBMS is additionally deployed on the Linux OS server for the Outlook add-in, add 2 GB RAM and 20 GB HDD.

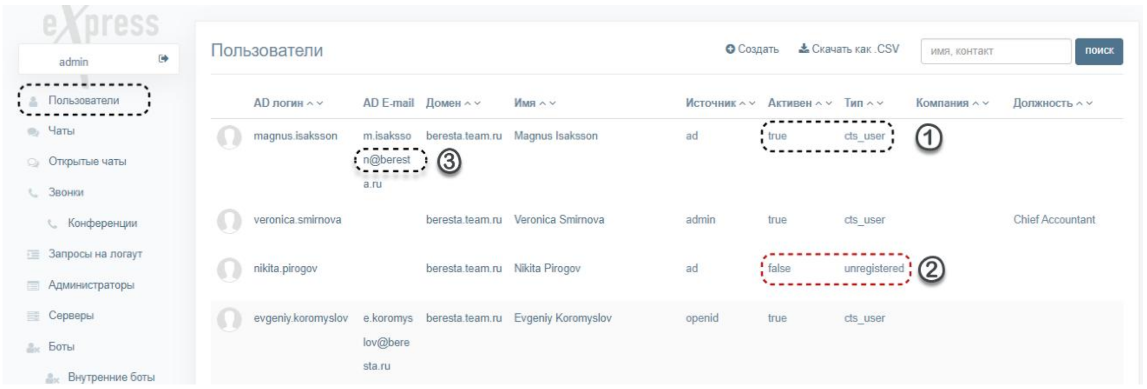
If there are no other services on the virtual machine, the PostgreSQL DBMS server running Windows OS must meet the following technical requirements (Table 3):

Table 3

Component	**Parameters**
Processor	64-bit 1.4 GHz processor 2 core
RAM	2 GB
Operating System	Windows Server 2016 or higher
Hard Disk	40 GB or more

All users who want to use the add-in must be registered in the Express CS and be active. The user status can be checked in the Express CS administrator web interface – "Users" section (Figure 1).

Important! When using MS Exchange, the conference creator must be registered in the Express CS with the email address specified as their Primary SMTP address in MS Exchange.



Пользователи

AD логин	AD E-mail	Домен	Имя	Источник	Активен	Тип	Компания	Должность
magnus.isaksson	m.isaksson@beresta.ru	beresta.team.ru	Magnus Isaksson	ad	true	cts_user		
veronica.smirnova		beresta.team.ru	Veronica Smirnova	admin	true	cts_user		Chief Accountant
nikita.pirogov		beresta.team.ru	Nikita Pirogov	ad	false	unregistered		
evgeniy.koromyslov	e.koromyslov@beresta.ru	beresta.team.ru	Evgeniy Koromyslov	openid	true	cts_user		

Figure 1. User Status: 1 – Active; 2 – Inactive

For an upcoming conference to be automatically added in the Express CS application of the invited user, that user must be registered in the application with the same email address that the conference creator used to send the invitation via the add-in (Figure 1, indicator 3).

The user will receive an email with a direct join link if:

- the user is not present in the Express CS;
- the user has an "Inactive" status in the Express CS;
- the user is registered in the Express CS with a different email address.

Chapter 2

ARCHITECTURE

Different deployment schemes for the Outlook add-in server-side are used depending on the Express CS architectural solution employed. The most common (standard) architectural solutions are described below.

Specific deployment schemes are determined by customer requirements after consultation with the development company.

If a single email template is planned for use, all express-core-service instances can use one database. If different templates are required for different express-core-service instances, then a separate database and a separate express-template-service should be deployed for each of them.

SINGLE CTS OR SPLIT CTS

If Express CS contains a single CTS server, deploy the server part of the add-in in a single instance. The typical deployment scheme for this type is presented below (Figure 2):

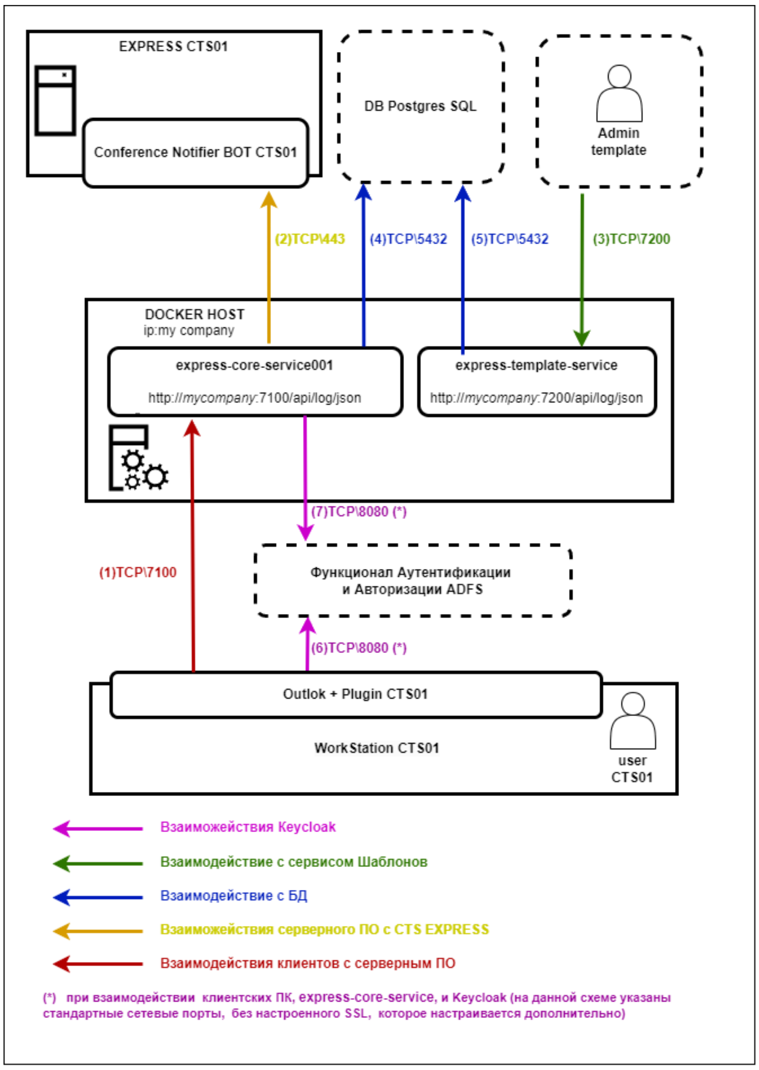

EXPRESS CTS01
Conference Notifier BOT CTS01
DB Postgres SQL
Admin template
(2)TCP\443
(4)TCP\5432
(5)TCP\5432
(3)TCP\7200
DOCKER HOST
ip:my company
express-core-service001
http://mycompany:7100/api/log/json
express-template-service
http://mycompany:7200/api/log/json
(7)TCP\8080 (*)
(1)TCP\7100
Функционал Аутентификации и Авторизации ADFS
(6)TCP\8080 (*)
Outlok + Plugin CTS01
WorkStation CTS01
user CTS01
Взаиможействия Keycloak
Взаимодействие с сервисом Шаблонов
Взаимодействие с БД
Взаиможействия серверного ПО с CTS EXPRESS
Взаимодействия клиентов с серверным ПО
(*) при взаимодействии клиентских ПК, express-core-service, и Keycloak (на данной схеме указаны стандартные сетевые порты, без настроенного SSL, которое настраивается дополнительно)

Figure 2. Typical Scheme: 1 CTS, 1 express-core-service, 1 express-template-service

The network interaction numbers correspond to the row number in Table 4 Appendix 2.

MULTIPLE CTS

If Express CS contains multiple CTS servers, deploy the server part of the add-in separately for each CTS server (deployment on a single Docker server, but with different ports and container names, or on multiple Docker servers is allowed). The typical deployment scheme for this type is presented below (Figure 3):

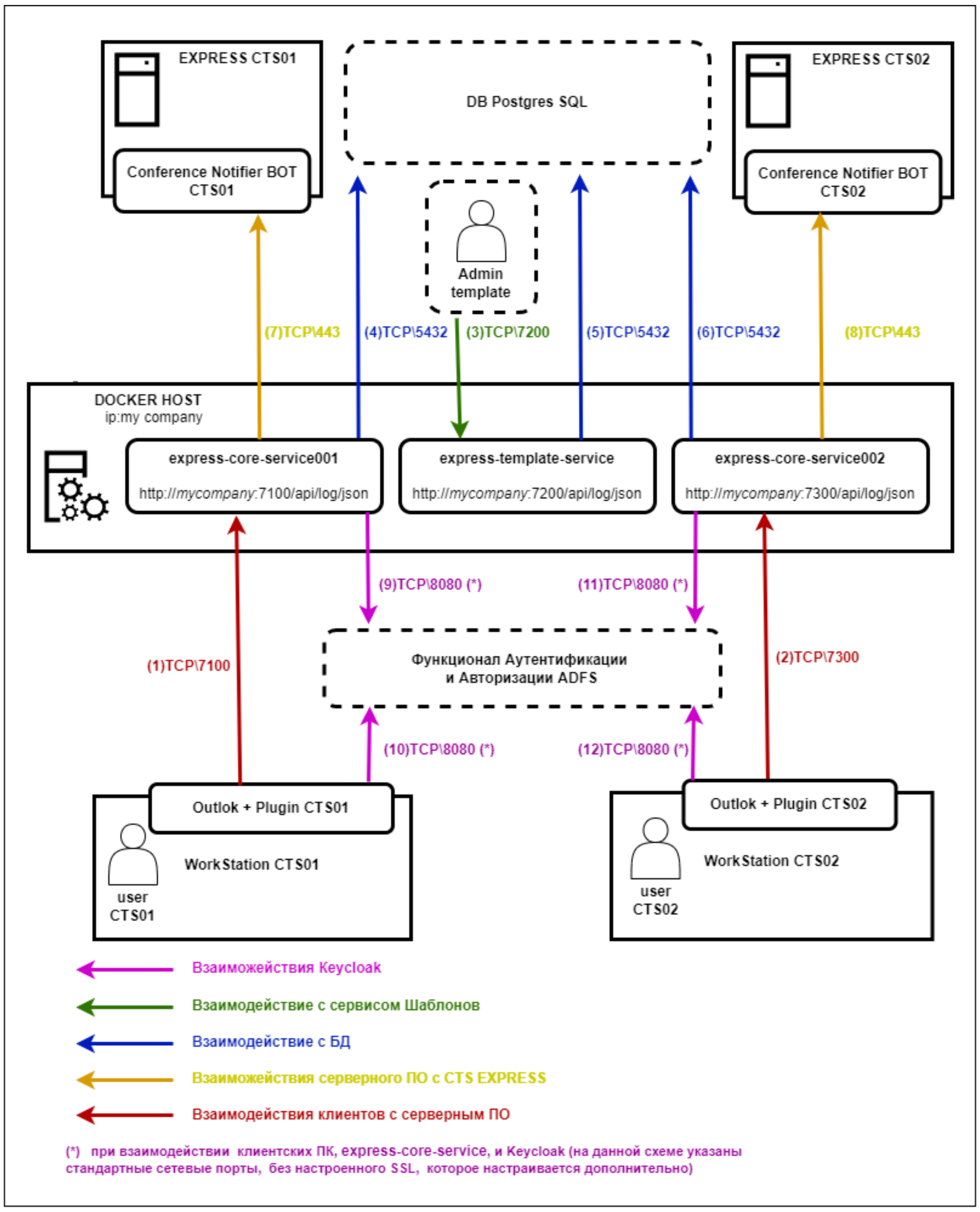

EXPRESS CTS01
Conference Notifier BOT CTS01
DB Postgres SQL
EXPRESS CTS02
Conference Notifier BOT CTS02
Admin template
(7)TCP\443
(4)TCP\5432
(3)TCP\7200
(5)TCP\5432
(6)TCP\5432
(8)TCP\443
DOCKER HOST
ip:my company
express-core-service001
http://mycompany:7100/api/log/json
express-template-service
http://mycompany:7200/api/log/json
express-core-service002
http://mycompany:7300/api/log/json
(9)TCP\8080 (*)
(11)TCP\8080 (*)
(1)TCP\7100
Функционал Аутентификации и Авторизации ADFS
(2)TCP\7300
(10)TCP\8080 (*)
(12)TCP\8080 (*)
Outlok + Plugin CTS01
WorkStation CTS01
user CTS01
Outlok + Plugin CTS02
WorkStation CTS02
user CTS02
Взаиможействия Keycloak
Взаимодействие с сервисом Шаблонов
Взаимодействие с БД
Взаиможействия серверного ПО с CTS EXPRESS
Взаимодействия клиентов с серверным ПО
(*) при взаимодействии клиентских ПК, express-core-service, и Keycloak (на данной схеме указаны стандартные сетевые порты, без настроенного SSL, которое настраивается дополнительно)

Figure 3. Typical Scheme: 2 CTS, 2 express-core-service, 1 express-template-service

The network interaction numbers correspond to the row number in Table 5 Appendix 2.

MULTIPLE CTS AND ETS

If Express CS contains multiple CTS servers unified by an ETS server, deploy the server part of the add-in only for the ETS server. The typical deployment scheme for this type is presented below (Figure 4):

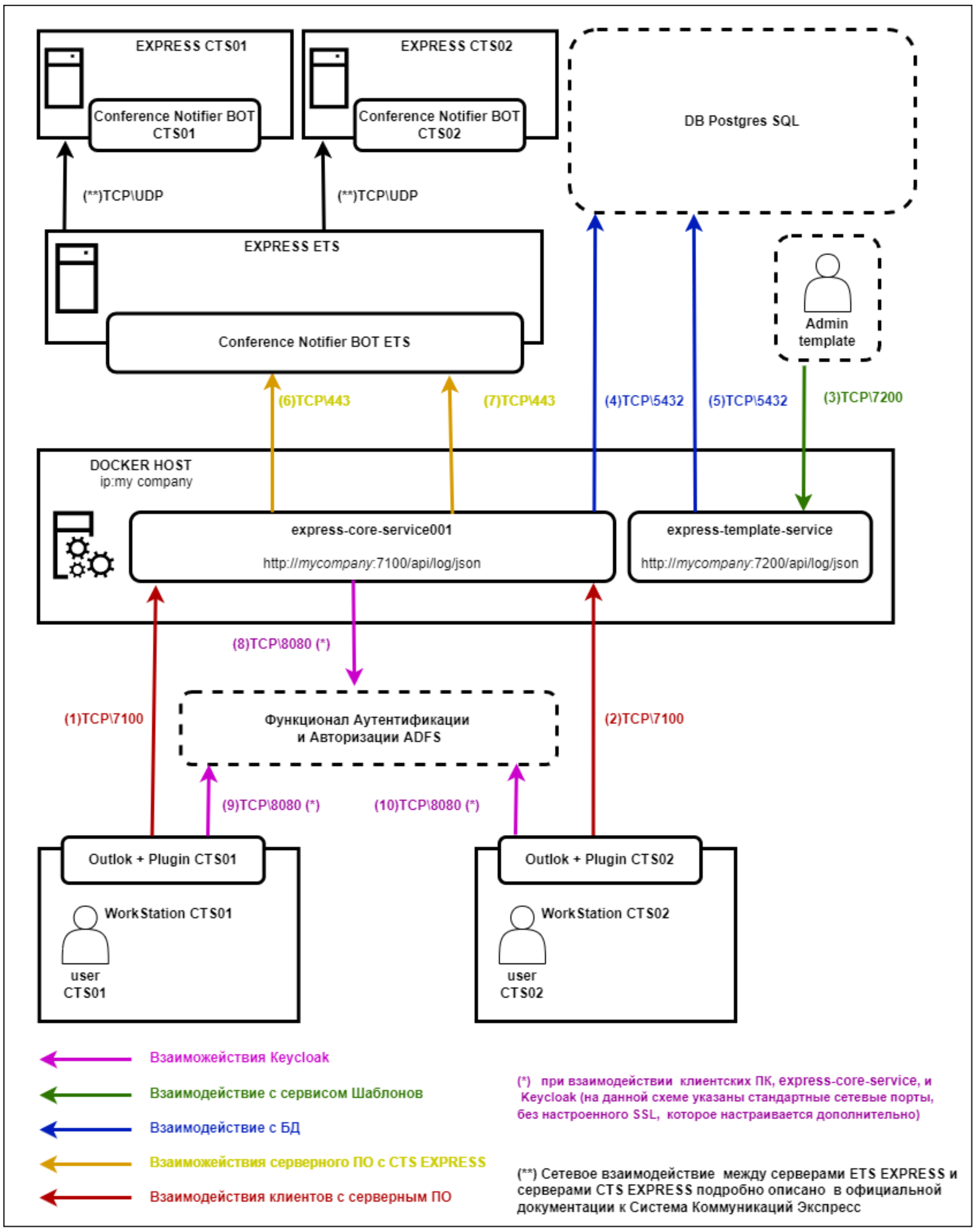

EXPRESS CTS01
Conference Notifier BOT CTS01
EXPRESS CTS02
Conference Notifier BOT CTS02
DB Postgres SQL
(**)TCP\UDP
(**)TCP\UDP
EXPRESS ETS
Conference Notifier BOT ETS
Admin template
(6)TCP\443
(7)TCP\443
(4)TCP\5432
(5)TCP\5432
(3)TCP\7200
DOCKER HOST
ip:my company
express-core-service001
http://mycompany:7100/api/log/json
express-template-service
http://mycompany:7200/api/log/json
(8)TCP\8080 (*)
(1)TCP\7100
Функционал Аутентификации и Авторизации ADFS
(2)TCP\7100
(9)TCP\8080 (*)
(10)TCP\8080 (*)
Outlok + Plugin CTS01
WorkStation CTS01
user CTS01
Outlok + Plugin CTS02
WorkStation CTS02
user CTS02
Взаиможействия Keycloak
Взаимодействие с сервисом Шаблонов
Взаимодействие с БД
Взаиможействия серверного ПО с CTS EXPRESS
Взаимодействия клиентов с серверным ПО
(*) при взаимодействии клиентских ПК, express-core-service, и Keycloak (на данной схеме указаны стандартные сетевые порты, без настроенного SSL, которое настраивается дополнительно)
(**) Сетевое взаимодействие между серверами ETS EXPRESS и серверами CTS EXPRESS подробно описано в официальной документации к Система Коммуникаций Экспресс

Figure 4. Typical Scheme: 1 ETS, 2 CTS, 1 express-core-service, 1 express-template-service

The network interaction numbers correspond to the row number in Table 6 Appendix 2Таблица3.

Chapter 3

INSTALLING AND CONFIGURING CONFERENCE NOTIFIER CHAT BOT

This section describes the procedure for connecting the Conference Notifier Bot chat bot API on the CTS server of the Express CS software solution.

Note. The procedure is described for Debian 11.4 and Ubuntu 2X. If issues arise while configuring the chat bot on other operating systems, it is recommended to contact the development company.

STEP 1. ENABLING CONFERENCE NOTIFIER BOT API WITHOUT PASSWORD

Important! When configuring the chat bot without protection, any user from the external network segment can use it if it is open.

Key points:

- if the Express CS contains multiple CTS servers, this procedure must be performed on each CTS that will interact with express-core-service;
- If the Express CS contains one or multiple ETS servers, this procedure must be performed on each ETS and CTS that will interact with express-core-service.

To enable the Conference Notifier Bot API without a password:

1. On the CTS server, add the following code to the settings.yamll[1] file:
   ```
   conference_bot_env_override:
     API_ENABLED: true
   ```
   In a split-server architecture, changes are made only to the file on the Back CTS server.
2. Save the file and run from /OPT/EXPRESS:
   ```
   dpl -d conference_bot
   ```
3. Open the administrator web interface and navigate to "Bots → Internal Bots → Conference Notifier Bot".

   The "Edit Bot" window will open.
4. In the opened window, select the checkbox in the "Enabled" field and click the "Save" button. Add port 4000 if necessary.
5. Check the chat bot's accessibility using a CURL request, for example:
   ```
   curl -X POST -H "Content-Type: application/json" -d
   '{"name":"test_plugin_001","members":["user001@mydomain.com","
   user002@mydomain.com"],"admins":["user001@mydomain.com
   "],"creator":"user001@mydomain.com","start_at":"2023-12-
   21T10:10:00.0Z","end_at":"2023-12-
   21T11:10:00.0Z","link":{"link_type":"public","access_code":null}}'
   https://my-CTS-domain.com/api/v1/conference_bot/conferences/
   ```
   Note. Parameters marked in red must be updated according to the specific request.

   If the chat bot is accessible, CURL will output:
   ```
   "status":"ok"
   ```
6. In the CTS server administrator web interface, navigate to "Calls → Conferences".
7. In the search field, enter test_plugin_001 (specified in step 5) and click "Search".

   If the table displays a conference named test_plugin_001, the chat bot is configured correctly.

[1] The presumed path is /opt/express/settings.yaml.

To check the Docker container status after installing the bot on the Express server, enter the command:

```
docker ps
```

The conference_bot container should have the statuses "Up" and "healthy".

Important! The versions of the conference_bot and messaging containers must match. If they do not match, the entire Express CS should be updated.

STEP 2. SECURING CONFERENCE NOTIFIER BOT

Key points:

- if the Express CS contains multiple CTS servers, this procedure must be performed on each CTS that will interact with express-core-service;
- if the Express CS contains a single unified ETS, this procedure should be performed only on it.

To secure the Conference Notifier Bot, perform the following bearer token setup:

1. Ensure that Express CS version 3.8 or later is installed. If the version is lower, update Express to the latest version.
2. Ensure that the distribution version of the "Express CS Add-in for Outlook" (server and client) is 1.2.0.0 or later.

   Attention! The entered credentials will be used when configuring the Outlook add-in.

3. Generate a BEARER TOKEN of 30 to 40 characters using a password generation tool or by using the following commands:

   ```
   openssl rand -hex 40
   ```

   or

   ```
   head /dev/urandom | tr -dc A-Za-z0-9 | head -c 40 ; echo
   ```

4. On the CTS server, add the following code to the conference_bot_env_override section in the settings.yaml file[2], using the BEARER TOKEN created previously:

   ```
     API_AUTH_METHOD: BEARER_TOKEN
     BEARER_TOKEN: 3bDewf52b3268sdg59f1f7fff33w01dd3c0431
   ```

   After which the conference_bot_env_override section should look like this:

   ```
   conference_bot_env_override:
     API_ENABLED: true
     API_AUTH_METHOD: BEARER_TOKEN
     BEARER_TOKEN: 3bDewf52b3268sdg59f1f7fff33w01dd3c0431
   ```

   In the case of a split server architecture, changes are made only to the file on the Back CTS server.

   The token value should be invented by you or generated by any third-party software (it must be at least 40 characters long and contain uppercase and lowercase Latin letters and numbers).

5. Save the file and run the following command from /OPT/EXPRESS:

   ```
   dpl -d conference_bot
   ```

6. Check the unavailability of the chatbot with a request without a bearer token using CURL, for example:

[2] Presumed path is /opt/express/settings.yaml

```
curl -X POST -H "Content-Type: application/json" -d
'{"name":"test_plugin_001","members":["user001@mydomain.com","
user002@mydomain.com"],"admins":["user001@mydomain.com
"],"creator":"user001@mydomain.com","start_at":"2023-12-
21T10:10:00.0Z","end_at":"2023-12-
21T11:10:00.0Z","link":{"link_type":"public","access_code":null}}'
https://my-CTS-domain.com/api/v1/conference_bot/conferences/
```

Note. Parameters marked in red must be updated according to the specific request.

If the chatbot is unavailable without a bearer token (which is expected), CURL will output an empty string.

7. Check the availability of the chatbot with a request with a bearer token using CURL, for example:

```
curl -X POST -H "Content-Type: application/json" -H "Authorization:
Bearer 3bfef52b32685srdsrhderhFGd1f73301dd3c0431" -d
'{"name":"test_plugin_001","members":["user001@mydomain.com","
user002@mydomain.com"],"admins":["user001@mydomain.com
"],"creator":"user001@mydomain.com","start_at":"2023-12-
21T10:10:00.0Z","end_at":"2023-12-
21T11:10:00.0Z","link":{"link_type":"public","access_code":null}}'
https://my-CTS-domain.com/api/v1/conference_bot/conferences/
```

Note. Parameters marked in red must be updated according to the specific request.

If the chatbot is available, CURL will output:

```
"status":"ok"
```

8. In the administrator web interface of the CTS server, go to the “Calls Conferences” section.
9. In the search field, enter test_plugin_001 (set in item 6) and click the “Search” button.
10. If a conference with the name test_plugin_001 is displayed in the table, the chatbot is configured correctly.

Chapter 4

INSTALLING THE OUTLOOK ADD-IN

To install the Outlook add-in, follow the steps described below. The distribution package (server and client parts) is available for download at the following link: https://nc.express.ms/s/f3iGiJieaGyPM7j?path=%2FADFS (select the current product version).

STEP 1. UPDATING THE SERVER PART (OPTIONAL)

This step is performed if the service was already deployed as part of the previous add-in version:

1. Upload new versions of the service image-images for express-meeting-core-service and express-meeting-template-service to Docker.
2. In the folders containing the ENV and YML files, run the command:

   ```
   docker-compose --env-file ./.env -f ./docker-compose.yml down
   ```

3. Take the new ENV and YML files from the new version of the distribution package and specify your parameters and the new images in them.
4. Replace the old ENV and YML files with the new ones in the corresponding folder on the Docker server.
5. Execute "Step 2. Preparing the Database" of the current chapter to update the DB structure.
6. Run the containers from the docker folder:

   ```
   docker-compose --env-file ./.env -f ./docker-compose.yml up -d
   ```

STEP 2. PREPARING THE DATABASE

To deploy the DB that stores the invitation block templates used when creating an invitation in an Outlook email:

1. Go to the Windows machine that has access to the PostgreSQL server. The machine must have .NET Desktop Runtime 7.0 (https://dotnet.microsoft.com/en-us/download/dotnet/7.0).
2. Transfer the "tools" folder from the distribution package to the machine.
3. In the tools\appsettings.json file, set the PostgreSQL DB connection string in the key:

   ```
   "Connection":"Host=127.0.0.1;Port=5432;Database=express_meeting_db;U
   sername=db_express;Password=Pass1234;Pooling=true;Minimum Pool
   Size=50;Maximum Pool Size=100;Include Error Detail=True;"
   ```

   Note. The parameters Username=db_express and Password=Pass1234 must correspond to the credentials that were created during PostgreSQL configuration.

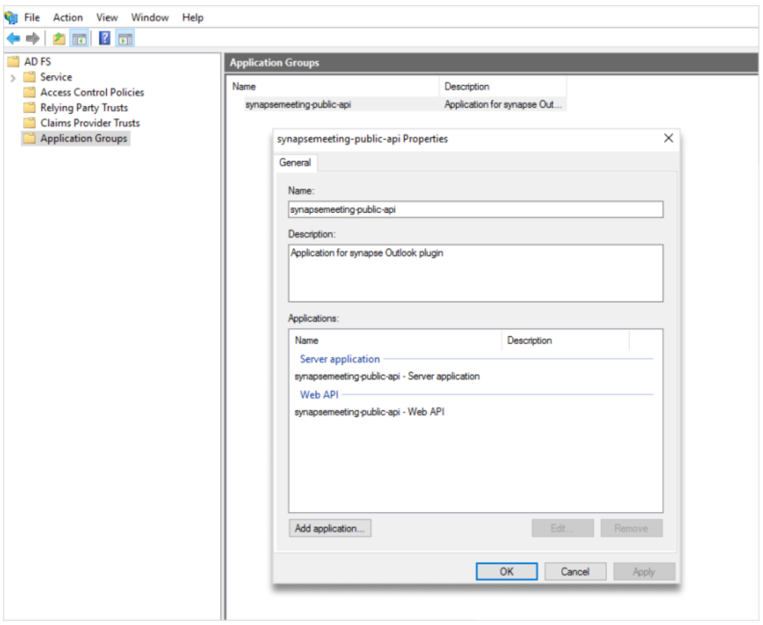
   Save the changes.
4. Run the following from the "tools" folder in the console:

   ```
   ExpressMeeting.Tools.TemplatesDB.exe --ef-migrate
   ```

5. Wait for the utility to complete its work.

   The DB is ready for use.

1. In AD FS, configure the Application Group for plugin request authorization (Figure 5):



Name	Description
synapsemeeting-public-api	Application for synapse Out...

synapsemeeting-public-api Properties

Name: synapsemeeting-public-api

Description: Application for synapse Outlook plugin

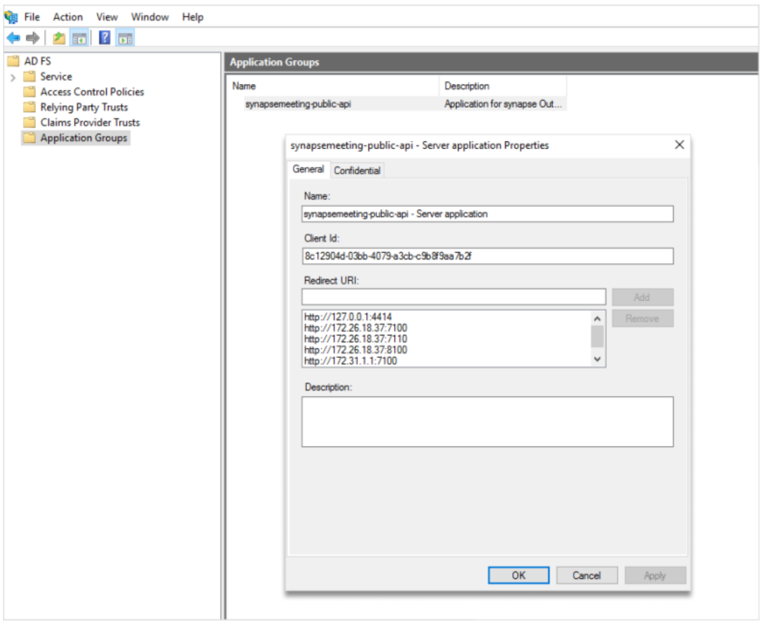
Applications:

Name	Description
Server application	
synapsemeeting-public-api - Server application	
Web API	
synapsemeeting-public-api - Web API	

Figure 5. Application Group Settings

2. Server application settings (Figure 6, Figure 7):

- in the Redirect URI section, the address of the core-service must be specified (service deployment is described in the section "Step 4. Deploying express-core-service and template-core-service in Docker"):



Name	Description
synapsemeeting-public-api	Application for synapse Out...

synapsemeeting-public-api - Server application Properties

Name: synapsemeeting-public-api - Server application

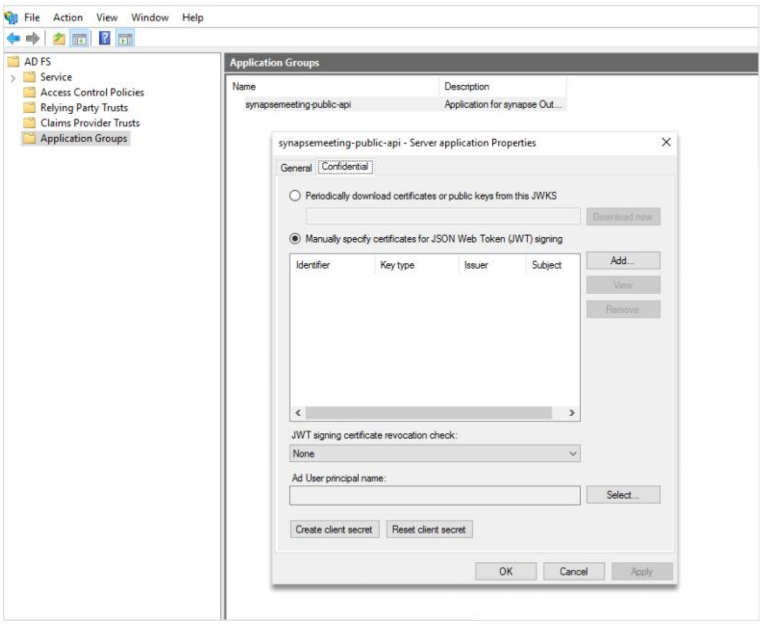
Client Id: 8c12904d-03bb-4079-a3cb-c9b8f9aa7b2f

Redirect URI:

```
http://127.0.0.1:4414
http://172.26.18.37:7100
http://172.26.18.37:7110
http://172.26.18.37:8100
http://172.31.1.1:7100
```

Description:

Figure 6. Server application settings (a)



File Action View Window Help

AD FS
- Service
- Access Control Policies
- Relying Party Trusts
- Claims Provider Trusts
- Application Groups

Application Groups

Name	Description
synapsemeeting-public-api	Application for synapse Out...

synapsemeeting-public-api - Server application Properties

General | Confidential

( ) Periodically download certificates or public keys from this JWKS — Download now

(●) Manually specify certificates for JSON Web Token (JWT) signing

Identifier	Key type	Issuer	Subject

Add... View Remove

JWT signing certificate revocation check: None

Ad User principal name: Select...

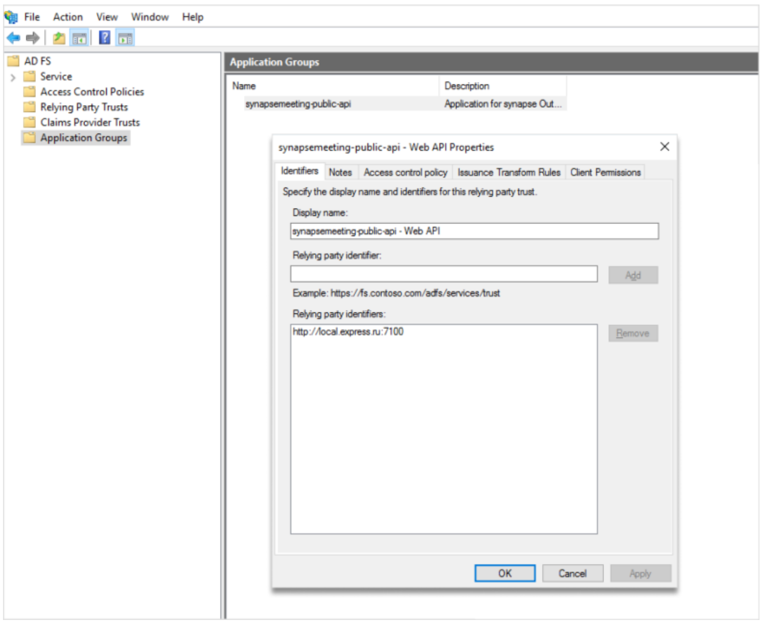
Create client secret | Reset client secret

OK Cancel Apply

Figure 7. Server application settings (b)

- when creating, fill in the Client Secret, Client ID, and port for the Redirect URI (http://127.0.0.1:4414. 127.0.0.1 is a mandatory address, you can use any other port that will be open on the client machine where Outlook itself is located). These parameters will later be used to configure the authorization of requests from the add-in and the express-core-service service.

3. Web API settings (Figure 8 – Figure 11):



File Action View Window Help

AD FS
- Service
- Access Control Policies
- Relying Party Trusts
- Claims Provider Trusts
- Application Groups

Application Groups

Name	Description
synapsemeeting-public-api	Application for synapse Out...

synapsemeeting-public-api - Web API Properties

Identifiers | Notes | Access control policy | Issuance Transform Rules | Client Permissions

Specify the display name and identifiers for this relying party trust.

Display name: synapsemeeting-public-api - Web API

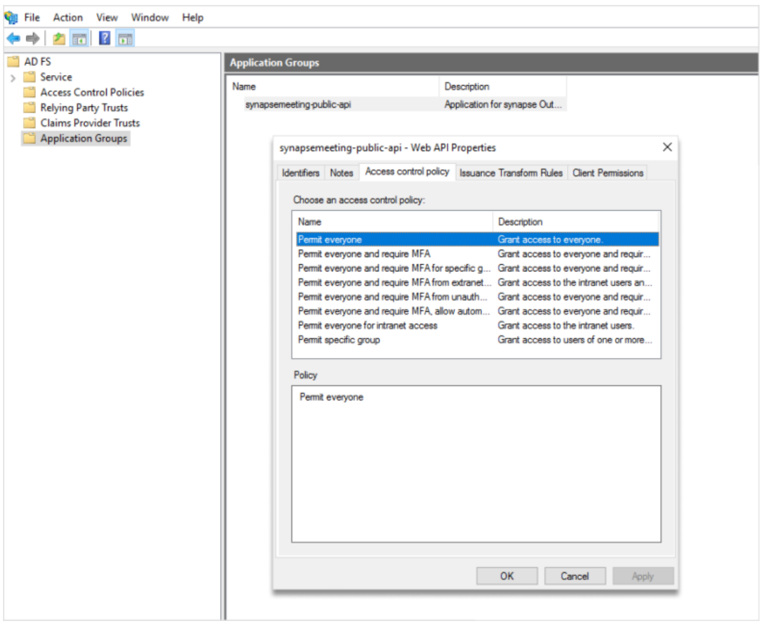
Relying party identifier: — Add

Example: https://fs.contoso.com/adfs/services/trust

Relying party identifiers: http://local.express.ru:7100 — Remove

OK Cancel Apply

Figure 8. Web API settings (a)



Name	Description
synapsemeeting-public-api	Application for synapse Out...

synapsemeeting-public-api - Web API Properties

Choose an access control policy:

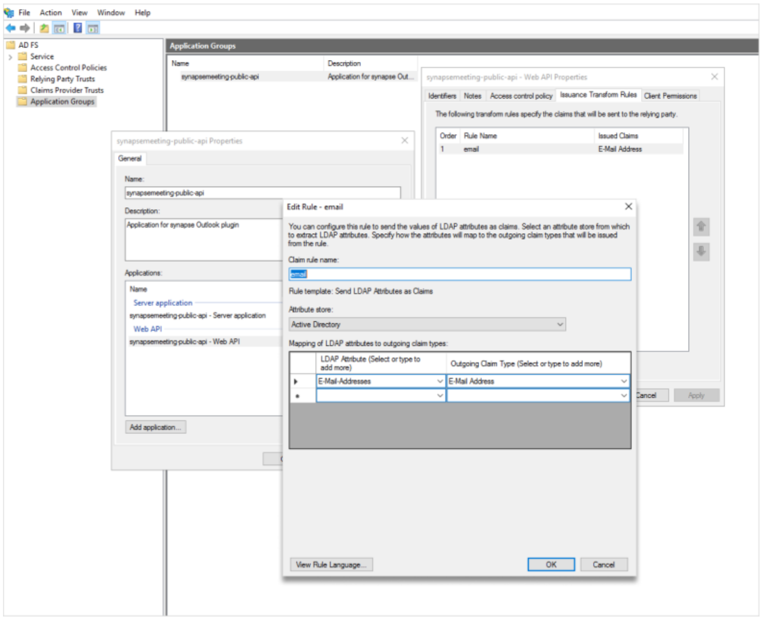
Name	Description
Permit everyone	Grant access to everyone.
Permit everyone and require MFA	Grant access to everyone and requir...
Permit everyone and require MFA for specific g...	Grant access to everyone and requir...
Permit everyone and require MFA from extranet...	Grant access to the intranet users an...
Permit everyone and require MFA from unauth...	Grant access to everyone and requir...
Permit everyone and require MFA, allow autom...	Grant access to everyone and requir...
Permit everyone for intranet access	Grant access to the intranet users.
Permit specific group	Grant access to users of one or more...

Policy

Permit everyone

Figure 9. Web API settings (b)

- add a rule to pass the email in the access token, as when creating a conference, a check is performed to match the email address from the token with the email address on whose behalf the conference is created.



Name	Description
synapsemeeting-public-api	Application for synapse Out...

synapsemeeting-public-api - Web API Properties

The following transform rules specify the claims that will be sent to the relying party.

Order	Rule Name	Issued Claims
1	email	E-Mail Address

synapsemeeting-public-api Properties

Name: synapsemeeting-public-api

Description: Application for synapse Outlook plugin

Applications:

Name
Server application
synapsemeeting-public-api - Server application
Web API
synapsemeeting-public-api - Web API

Edit Rule - email

You can configure this rule to send the values of LDAP attributes as claims. Select an attribute store from which to extract LDAP attributes. Specify how the attributes will map to the outgoing claim types that will be issued from the rule.

Claim rule name: email

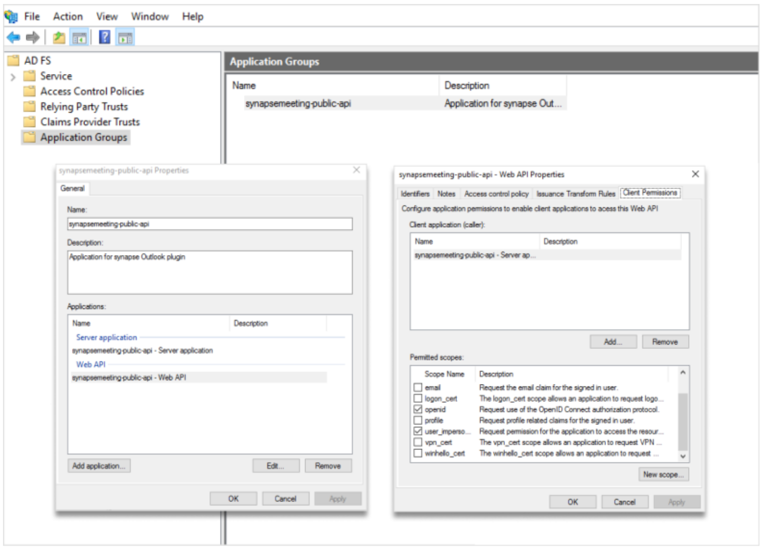
Rule template: Send LDAP Attributes as Claims

Attribute store: Active Directory

Mapping of LDAP attributes to outgoing claim types:

LDAP Attribute (Select or type to add more)	Outgoing Claim Type (Select or type to add more)
E-Mail-Addresses	E-Mail Address

Figure 10. Web API settings (c)



Application Groups

Name	Description
synapsemeeting-public-api	Application for synapse Out...

synapsemeeting-public-api Properties

Name: synapsemeeting-public-api

Description: Application for synapse Outlook plugin

Applications:

Name	Description
Server application	
synapsemeeting-public-api - Server application	
Web API	
synapsemeeting-public-api - Web API	

synapsemeeting-public-api - Web API Properties

Configure application permissions to enable client applications to acess this Web API

Client application (caller):

Name	Description
synapsemeeting-public-api - Server ap...	

Permitted scopes:

	Scope Name	Description
☐	email	Request the email claim for the signed in user.
☐	logon_cert	The logon_cert scope allows an application to request logo...
☑	openid	Request use of the OpenID Connect authorization protocol.
☐	profile	Request profile related claims for the signed in user.
☑	user_imperso...	Request permission for the application to access the resour...
☐	vpn_cert	The vpn_cert scope allows an application to request VPN ...
☐	winhello_cert	The winhello_cert scope allows an application to request ...

Figure 11. Web API settings (d)

- add the URL specified on the Identifiers tab to the Relying party identifier field. Its value will be used to configure the authorization of requests from the plugin and the express-core-service service.

STEP 4. DEPLOYING EXPRESS-CORE-SERVICE AND TEMPLATE-CORE-SERVICE IN DOCKER

The service is deployed in Docker.

To deploy the service:

1. Load the service image files into Docker from the images folder:

```
docker load --input express-meeting-core-service-main.tar
docker load --input express-meeting-template-service-main.tar
```

2. Prepare the service startup settings in the docker\.env file.

   The file contains the following settings:

   - the name of the image from which the core-service container starts (do not change):

   ```
   EXPRESS_MEETING_CORE_SERVICE_MAIN_IMAGE_NAME=express-meeting-core-service-main
   ```

   - the port through which core-service will be accessible:

   ```
   EXPRESS_MEETING_CORE_SERVICE_PORT=7100
   ```

   - the name of the image from which the template-service container starts (do not change):

   ```
   EXPRESS_MEETING_TEMPLATE_SERVICE_MAIN_IMAGE_NAME=express-meeting-template-service-main
   ```

   - the port through which template-service will be accessible:

   ```
   EXPRESS_MEETING_TEMPLATE_SERVICE_PORT=7200
   ```

   - the IP address corresponding to the DNS name of the Keycloak server (specify actual data):

   ```
   EXPRESS_MEETING_EXTRA_HOST=api.bot.express:127.0.0.1
   ```

   - the IP address corresponding to the DNS name of the Keycloak server (specify actual data):

   ```
   EXPRESS_MEETING_EXTRA_HOST2=adfs.mycompany:127.0.0.1
   ```

   - the name of the file with core-service settings (do not change):

   ```
   EXPRESS_MEETING_CORE_ENV_FILE=core.env
   ```

   - the name of the file with template-service settings (do not change):

```
EXPRESS_MEETING_TEMPLATE_ENV_FILE=template.env
```

- the container folder for service logs (do not change):

```
EXPRESS_MEETING_DOCKER_LOGS_PATH=/public/express-meeting-logs
```

- the Docker host folder where service logs will be added from the containers (change to the required path if necessary):

```
EXPRESS_MEETING_LOGS_PATH=/public/express-outlook/logs
```

3. Prepare the core-service startup settings in the docker\core.env file.

   The file contains the following settings:

   - the address of the Conference Notifier Bot API (specify the actual address where the API is deployed):

   ```
   ExpressOptions__Uri=https://api.bot.express/api/v1/conference_bot
   /conferences/
   ```

   - the authorization scheme for the Conference Notifier Bot API (Accepted values: Basic – login/password authorization; StaticToken – static token authorization)

   ```
   ExpressOptions__AuthenticationScheme=StaticToken
   ```

   - the token that will be used when accessing the Conference Notifier Bot API (used if the ExpressOptions__AuthenticationScheme parameter is set to StaticToken):

   ```
   ExpressOptions__Token=Bearer 3bb326abc5609fed301dd3c0431
   ```

   - the login of the account used to access the Conference Notifier Bot API (used if the ExpressOptions__AuthenticationScheme parameter is set to Basic):

   ```
   ExpressOptions__UserLogin=admin
   ```

   - the password of the account used to access the Conference Notifier Bot API (used if the ExpressOptions__AuthenticationScheme parameter is set to Basic):

   ```
   ExpressOptions__UserPassword=admin
   ```

   - the authorization type for core-service (do not change):

   ```
   AuthorizationOptions__AuthType=adfs
   ```

   - the ADFS address (specify actual):

   ```
   AuthorizationOptions__Configuration__Url=https://adfs.mycompany
   ```

   - the Client ID value, which was obtained during AD FS preparation described in "Step 3. Preparing AD FS":):

   ```
   AuthorizationOptions__Configuration__ServiceClientId=00000000-
   0000-0000-0000-000000000000
   ```

   - the Client Secret value, which was obtained during AD FS preparation described in "Step 3. Preparing AD FS":

   ```
   AuthorizationOptions__Configuration__ServiceClientSecret=00000000
   00000000000000000000000000000000
   ```

   - the URL value, which was added to the Relying party identifier field during AD FS preparation described in "Step 3. Preparing AD FS":

   ```
   AuthorizationOptions__Configuration__Resource=http://local.expres
   s.ru:7100
   ```

   - the PostgreSQL DB connection string (this is the DB that stores the invitation block templates when creating an invitation in an Outlook email):

   ```
   DatabaseOptions__Connection=Host=127.0.0.1;Port=5432;Database=exp
   ress_meeting_db;Username=db_express;Password=Pass1234;Pooling=tru
   e;Minimum Pool Size=50;Maximum Pool Size=100;Include Error
   Detail=True;
   ```

   Note. The parameters "Username=db_express" and "Password=Pass1234" must correspond to the credentials that were created during PostgreSQL configuration.

- the folder in the core-service container where logs will be saved (do not change):

```
Serilog__WriteTo__1__Args__configureLogger__WriteTo__0__Args__pat
h=/public/express-meeting-logs/coreservise.log
```

- the folder in the core-service container where logs received from clients will be saved (do not change):

```
Serilog__WriteTo__2__Args__configureLogger__WriteTo__0__Args__pat
h=/public/express-meeting-logs/client/plugin.log
```

- the service runtime environment (do not change):

```
ASPNETCORE_ENVIRONMENT=Production
```

4. Preparation of template-service launch settings in the docker\template.env file.

   The file contains the following settings:

   - the PostgreSQL DB connection string (this is the DB that stores the invitation block templates when creating an invitation in an Outlook email):

   ```
   DatabaseOptions__Connection=Host=127.0.0.1;Port=5432;Database=exp
   ress_meeting_db;Username=db_express;Password=Pass1234;Pooling=tru
   e;Minimum Pool Size=50;Maximum Pool Size=100;Include Error
   Detail=True;
   ```

   Note. The parameters "Username=db_express" and "Password=Pass1234" must correspond to the credentials that were created during PostgreSQL configuration.

   - an arbitrary set of characters used to generate the key with which the JWT access token to the service will be signed:

   ```
   JWTOptions__Key=JFRINXV0LGIhLXlEVzlYdXlae316UU0zJzR9WVUtIWQ
   ```

   - the “Issuer” field value when generating and validating the JWT token:

   ```
   JWTOptions__Issuer=http://127.0.0.1
   ```

   - the “Audience” field value when generating and validating the JWT token:

   ```
   JWTOptions__Audience=http://127.0.0.1
   ```

   - the folder in the template-service container where logs will be saved (do not change):

   ```
   Serilog__WriteTo__1__Args__path=/public/express-meeting-
   logs/templateservise.log
   ```

   - the service runtime environment (do not change):

   ```
   ASPNETCORE_ENVIRONMENT=Production
   ```

5. Run the docker-containers from the folder:

   ```
   docker-compose --env-file ./.env -f ./docker-compose.yml up -d
   ```

   After launching the containers, if HTTPS addresses were specified in the ExpressOptions__Uri and AuthorizationOptions__Configuration__Url keys in the docker\core.env file, in case of necessity (for example, the certification authority for the company's certificates is unavailable), add the full certificate chain for the corresponding HTTPS connections to the express-meeting-core-service-main container. An example of adding certificates to the container is described in Appendix 1.

6. If necessary, add files from the following locations to the antivirus exceptions:

   ```
   /var/lib/docker
   ```

After launching the template-service container, navigating to the address served by this container at the path /front (e.g., http://localhost:7200/front) in the browser should open the login page for the add-in template administration system (Figure 12).

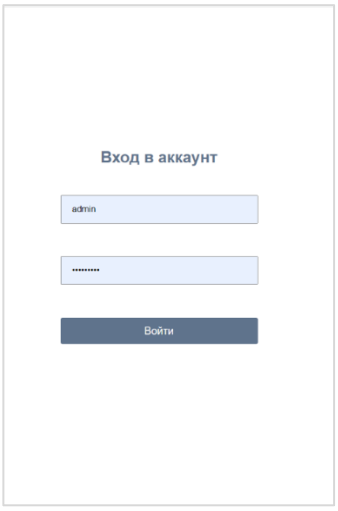


Figure 12

Note. The first login is performed using the admin account with an empty password.

STEP 5. DEPLOYING OUTLOOK ADD-IN ON CLIENT PCS

If the service was already deployed as part of the previous add-in version, then uninstall the previous version of the client software on the client machines and install the new version.

Important! After installing or updating the client part of the add-in on PCs running Windows OS, restarting MS Outlook is required.

To deploy the Outlook add-in:

1. Launch the ExpressMeeting_v_3.1.0.2.msi installer on the machine where Outlook is installed.
2. After installation, perform the following settings in the ExpressMeeting.dll.config[3] file for the add-in to work:
   - Address of the core-service (server part of the add-in) in the key (replace mycompany with the IP/DNS name of the Docker host machine where the core-service is deployed):

     ```
     <add key="ExpressMeetingUrl" value= "http://mycompany:7100/api/"
     />
     ```

     Note. The parameter can be initialized automatically during msi package installation. To do this, the ExpressMeetingUrl parameter must be passed in the command line.

     Example: ExpressMeeting_v_3.1.0.2.msi ExpressMeetingUrl="http://mycompany:7100/api/"

   - Timeout in seconds for establishing a connection when accessing ADFS:

     ```
     <add key="Timeout" value="120"/>
     ```

     Note. The parameter can be initialized automatically during msi package installation. To do this, the DenyLinkType parameter must be passed in the command line.

     Example: ExpressMeeting_v_3.1.0.2.msi DenyLinkType=true

   - If necessary, set the mode to prohibit conference type selection. The dropdown list for type selection (Public/Corporate/Trusted)[4] will be hidden in the meeting

[3] The presumed path is c:\Program Files\Express\ExpressMeeting\ExpressMeeting.dll.config.

[4] More details on the types of conferences created can be found in the Express CS user guides

settings. Only Public meetings will always be created. To do this, add the following parameter to the configuration/appSettings section:

```
<add key="DenyLinkType" value="true"/>
```

Note. The parameter can be initialized automatically during msi package installation. To do this, the DenyLinkType parameter must be passed in the command line.

Example: ExpressMeeting_v_3.1.0.2.msi DenyLinkType=true

- If necessary, set the type of conference, created by default (possible values: Public, Trusts, Corporate). Meetings of the specified type will be created when the Create Conference button is clicked. To do this, add the following parameter to the configuration/appSettings section:

```
<add key="DefaultLinkType" value="Corporate"/>
```

Note. The parameter can be initialized automatically during msi package installation. To do this, the DefaultLinkType parameter must be passed in the command line.

Example: ExpressMeeting_v_3.1.0.2.msi DefaultLinkType=Corporate

- If necessary, in the ExpressMeeting.dll.config file, set the minimum level of password requirements (possible values: Public, Trusts, Corporate). The password switch will be unavailable for conferences of the specified type, as well as more public ones, when clicking the Create Conference button. To do this, add the following parameter to the configuration/appSettings section:

```
<add key="LinkMinimumLevelRequirePassword" value="Corporate"/>
```

Note. The parameter can be initialized automatically during msi package installation. To do this, the LinkMinimumLevelRequirePassword parameter must be passed in the command line.

Example: ExpressMeeting_v_3.1.0.2.msi LinkMinimumLevelRequirePassword=Corporate

- If necessary, enable background authentication token renewal so the user does not have to re-enter credentials after prolonged inactivity. To do this, add the following parameter to the configuration/appSettings section (Default value is false (background renewal is disabled)):

```
<add key="BackgroundRefreshTokenProcess" value="true"/>
```

Note. The parameter can be initialized automatically during msi package installation. To do this, the BackgroundRefreshTokenProcess parameter must be passed in the command line.

Example: ExpressMeeting_v_3.1.0.2.msi BackgroundRefreshTokenProcess=true.

- If necessary, set the background authentication token renewal interval. The period is specified in "hours:minutes:seconds" format. For example, for 15 minutes, specify "00:15:00". To do this, add the following parameter to the configuration/appSettings section:

```
<add key="BackgroundRefreshTokenProcessPeriod" value="00:15:00"/>
```

Note. The parameter can be initialized automatically during msi package installation. To do this, the BackgroundRefreshTokenProcessPeriod parameter must be passed in the command line.

Example: ExpressMeeting_v_3.1.0.2.msi BackgroundRefreshTokenProcessPeriod=00:15:00

- If it is necessary to enable Serilog internal logging (e.g., for diagnosing cases where main logs are not recorded), specify the full file path. To do this, add the following parameter to the configuration/appSettings section:

```
<add key="selfLogPath" value="C:\logs\serilogInternal.log"/>
```

3. In the serilogSettings.json[5] file set the core-service address (server part of the add-in) in the key (replace mycompany with the IP/DNS name of the Docker host machine where the core-service is deployed):

```
Serilog.WriteTo[Name=Telemetry].Args.telemetryUrl="http://mycompany:
7100/api/log/json"
```

Note. If the ExpressMeetingUrl parameter was passed in the command line when installing the msi-package, this parameter will be initialized automatically.

4. The add-in supports different logging levels. To change the logging level in the serilogSettings.json[6] file, set the value of the Serilog.MinimumLevel.Default key to the required level. Possible logging level values are: Verbose, Debug, Information, Warning, Error, Fatal. Where Verbose means log everything, Debug means log Debug and above, Information means log Information and above, etc.

Note. The parameter can be initialized automatically during msi package installation. To do this, the LogLevel parameter must be passed in the command line.

Example: ExpressMeeting_v_3.1.0.2.msi LogLevel=Information.

5. If necessary, add the following file locations to the antivirus exclusions on client PCs:
   - C:\Program Files\eXpress\ExpressMeeting (this path may vary for branded versions of the add-in);
   - C:\logs (if a different path was specified in Chapter 4, Step 5, item 2, then that path should be specified).

[5] The presumed path is c:\Program Files\Express\ExpressMeeting\serilogSettings.json.

[6] The presumed path is c:\Program Files\Express\ExpressMeeting\serilogSettings.json.

Chapter 5

ADMINISTRATOR WEB INTERFACE

This section describes the administrator web interface and how to work with it.

AUTHORIZING IN ADMINISTRATOR WEB INTERFACE

After installing the server part of the Outlook add-in, the login page for the add-in template administration system should open in the browser (Figure 13).

To authorize in the administrator console:

1. In the browser's address bar, enter the address of the administrator web interface (e.g., http://localhost:7200/front).

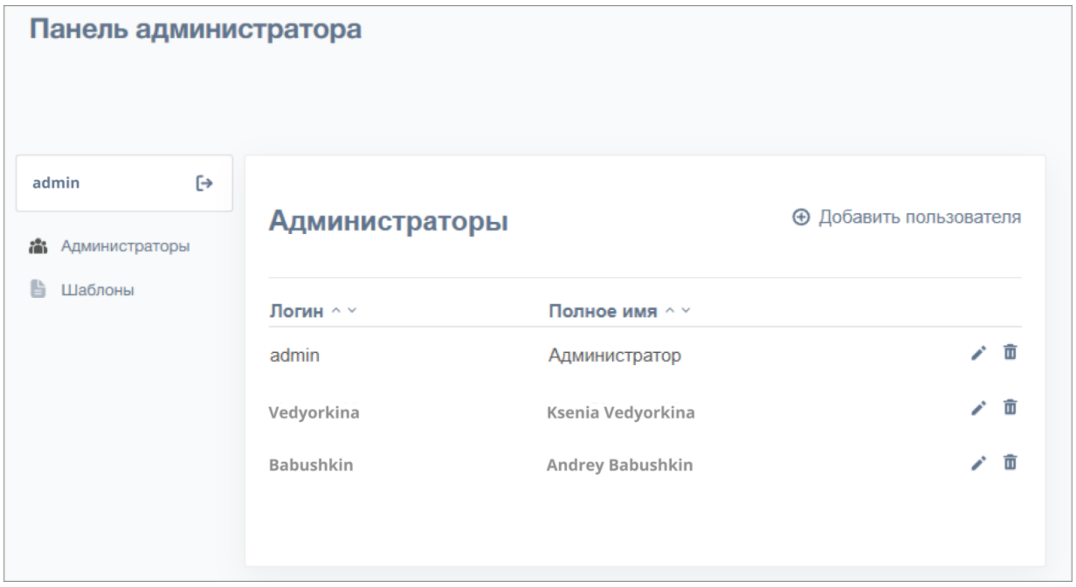
   The authorization window will open (Figure 14).
2. Enter the user account name and password into the corresponding fields.

   Note. The first login is performed using the admin account with an empty password.
3. Click the "Log in" button.

   Authorization will be granted.

DESCRIPTION OF THE ADMINISTRATOR PANEL INTERFACE

This section describes the administrator panel interface using the "Administrators" section interface as an example (Figure 13).



Логин	Полное имя
admin	Администратор
Vedyorkina	Ksenia Vedyorkina
Babushkin	Andrey Babushkin

Figure 13. "Administrators" Section Interface

MANAGING ADMINISTRATORS

The following operations are available to the administrator:

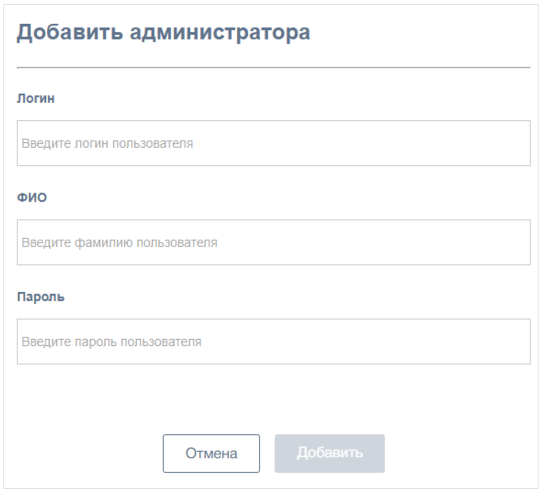
- adding an add-in administrator;
- editing add-in administrator data;
- deleting an add-in administrator.

To add an add-in administrator:

1. Click the "⊕ Add User" button.

   The administrator addition form will open (Figure 14).

2. Enter all necessary information in the form fields and click the "Add" button.



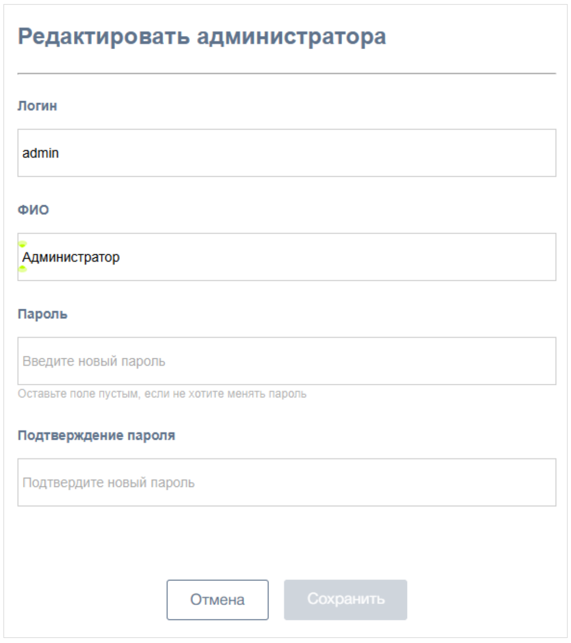
Добавить администратора

Логин	Введите логин пользователя
ФИО	Введите фамилию пользователя
Пароль	Введите пароль пользователя

Figure 14. Adding an administrator

To edit administrator data:

1. Select an administrator from the list (Figure 13) and click the "Edit" button. The administrator data editing form will open (Figure 15):



Редактировать администратора

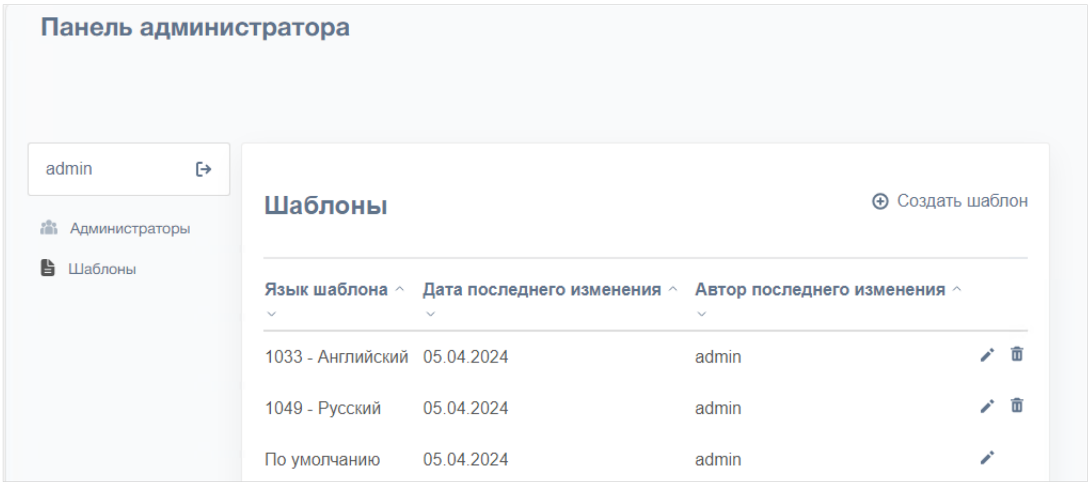
Логин	admin
ФИО	Администратор
Пароль	Введите новый пароль
	Оставьте поле пустым, если не хотите менять пароль
Подтверждение пароля	Подтвердите новый пароль

Figure 15. Making changes to administrator data

2. Make the necessary changes and click the "Save" button.

To delete an administrator, select the administrator from the list (Figure 13) and click the button.

In the "Templates" section (Figure 16), the administrator can create, edit (including in HTML mode), and delete invitation email templates.



Панель администратора

admin

Администраторы
Шаблоны

Шаблоны

Создать шаблон

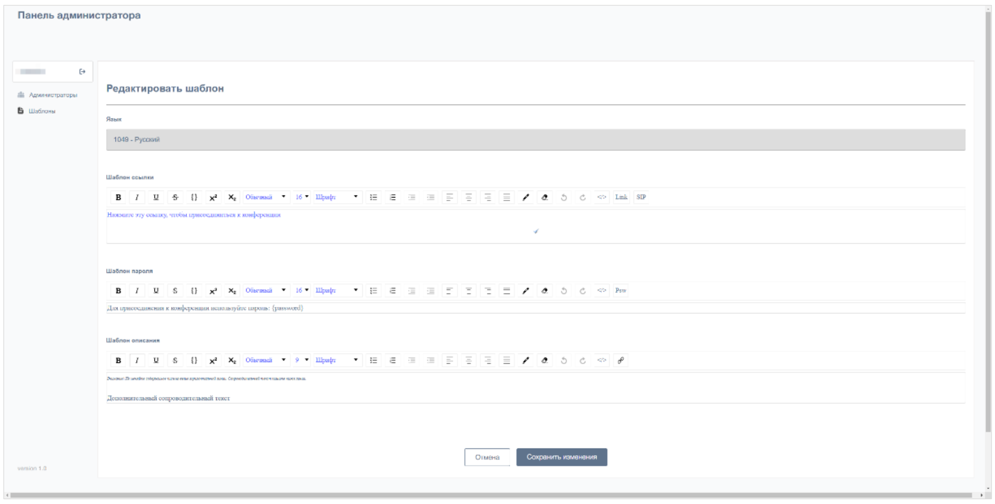
Язык шаблона	Дата последнего изменения	Автор последнего изменения	
1033 - Английский	05.04.2024	admin	✎ 🗑
1049 - Русский	05.04.2024	admin	✎ 🗑
По умолчанию	05.04.2024	admin	✎

Figure 16. The "Templates" section

To create a template:

1. Click the "Create Template" button

   A special visual template editor will open. It includes numerous functions: editing content, correcting font and text size, adding superscripts and subscripts, an HTML Editor, as well as buttons for inserting auto-substitution placeholders (conference links, current user's SIP numbers) (Figure 17).



Панель администратора

Администраторы
Шаблоны

Редактировать шаблон

Язык: 1049 - Русский

Шаблон ссылки: Нажмите эту ссылку, чтобы присоединиться к конференции

Шаблон пароля: Для присоединения к конференции используйте пароль: {password}

Шаблон описания: [illegible]
Дополнительный сопроводительный текст

Отмена | Сохранить изменения

version 1.0

Figure 17. Visual Template Editor

2. Fill in the template fields and click the "Save Changes" button.

To edit a template:

1. Select a template from the list (Figure 16) and click the ✎ button.

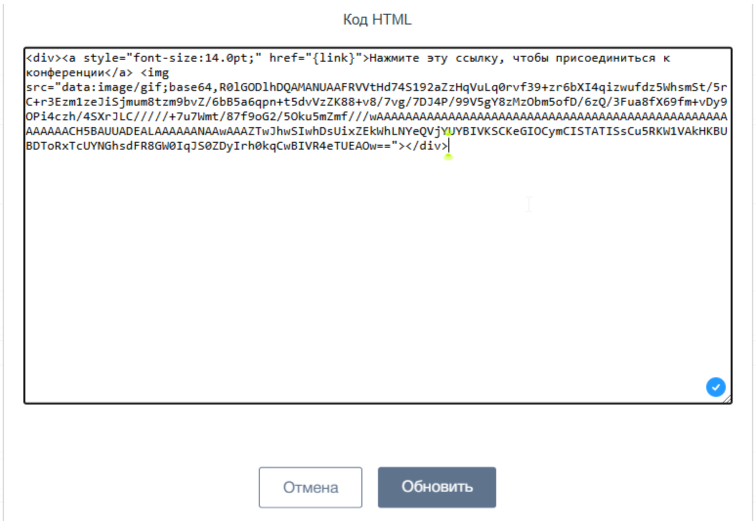
   The visual template editor will open (Figure 17).

2. Edit the template fields and click the "Save Changes" button.

To delete a template, select it from the list and click 🗑 .

HTML EDITOR

To switch to the HTML Editor, click the “</>” in visual template editor menu. The editor window will look as follows (Figure 18):



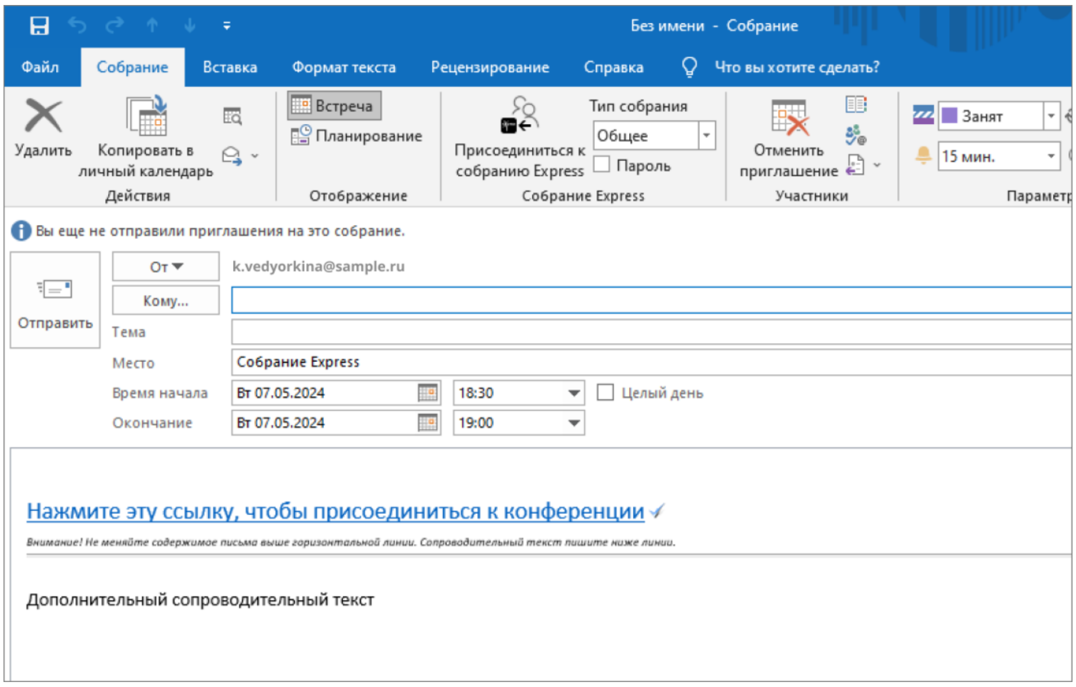
```
<div><a style="font-size:14.0pt;" href="{link}">Нажмите эту ссылку, чтобы присоединиться к
конференции</a> <img
src="data:image/gif;base64,R0lGODlhDQAMANUAAFRVVtHd74S192aZzHqVuLq0rvf39+zr6bXI4qizwufdz5WhsmSt/5r
C+r3Ezm1zeJiSjmum8tzm9bvZ/6bB5a6qpn+t5dvVzZK88+v8/7vg/7DJ4P/99V5gY8zMzObm5ofD/6zQ/3Fua8fX69fm+vDy9
OPi4czh/4SXrJLC/////+7u7Wmt/87f9oG2/5Oku5mZmf///wAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAACH5BAUUADEALAAAAAANAAwAAAZTwJhwSIwhDsUixZEkWhLNYeQVjYUYBIVKSCKeGIOCymCISTATISsCu5RKW1VAkHKBU
BDToRxTcUYNGhsdFR8GW0IqJS0ZDyIrh0kqCwBIVR4eTUEAOw=="></div>
```

Figure 18. HTML Editor

The editor fully supports HTML markup language. You can add images in it using the <img> tag and encode them in base64 format. For example:

```
<img src="data:image/gif;base64,R0lGODlhDQAMANUAAFRVVtHd74S192aZzHqVuLq0
rvf39+zr6bXI4qizwufdz5WhsmSt/5rC+r3Ezm1zeJiSjmum8tzm9bvZ/6bB5a6qpn+t5dvV
zZK88+v8/7vg/7DJ4P/99V5gY8zMzObm5ofD/6zQ/3Fua8fX69fm+vDy9OPi4czh/4SXrJLC
/////+7u7Wmt/87f9oG2/5Oku5mZmf///wAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAACH5BAUUADEALAAAAAANAAwAAAZTwJhwSIwhDsUixZEkWhLNYeQVjYUY
BIVKSCKeGIOCymCISTATISsCu5RKW1VAkHKBUBDToRxTcUYNGhsdFR8GW0IqJS0ZDyIrh0kq
CwBIVR4eTUEAOw==">
```

In the invitation email, this will be displayed as follows (Figure 19):



Вы еще не отправили приглашения на это собрание.

От: k.vedyorkina@sample.ru

Место: Собрание Express

Время начала: Вт 07.05.2024 18:30

Окончание: Вт 07.05.2024 19:00

Нажмите эту ссылку, чтобы присоединиться к конференции

Внимание! Не меняйте содержимое письма выше горизонтальной линии. Сопроводительный текст пишите ниже линии.

Дополнительный сопроводительный текст

Figure 19. Result of working in the HTML Editor

There are currently two templates—in Russian and English. The template is selected by the client part of the add-in and depends on the MS Outlook language locale. The Russian template will be automatically applied in Russian MS Outlook, and the English template in English MS Outlook. Manual template selection is not provided.

Appendix 1

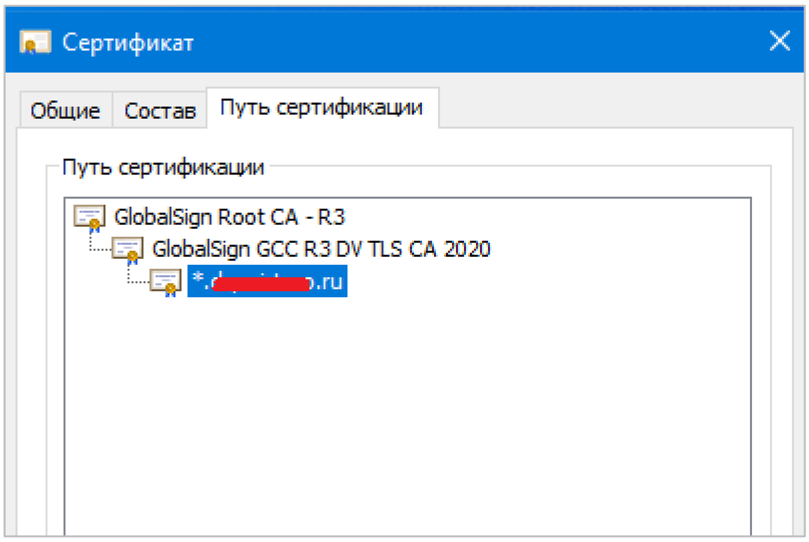
ADDING A CERTIFICATE TO THE EXPRESS-MEETING-CORE-SERVICE-MAIN CONTAINER

This section describes a typical solution for installing certificates into a Docker container. The procedure may vary depending on the infrastructure and architectural solution. This solution is typically used when the EXPRESS-MEETING-CORE-SERVICE container fails to establish an SSL connection with ADFS.

Important! For correct SSL operation, all certificates in the certificate chain (root certificate, intermediate certificates, end certificate) should be placed in the express-meeting-core-service-main container.

To add a certificate to the container:

1. Collect all certificates into chains as files in CRT format (Figure 20).



Сертификат

Общие | Состав | Путь сертификации

Путь сертификации

GlobalSign Root CA - R3
GlobalSign GCC R3 DV TLS CA 2020
*.[redacted].ru

Figure 20

2. On the Docker server where the express-meeting-core-service-main container is deployed, create a folder. Example folder name: /opt/express/bots/cert/.
3. Move the certificate files collected in step 1 to the folder created in step 2.
4. Grant the appropriate access permissions to the /opt/express/bots/cert/ folder.
5. In the docker-compose.yml file, add the following parameter to the volumes section for the express-meeting-core-service-main container parameters (Figure 21)

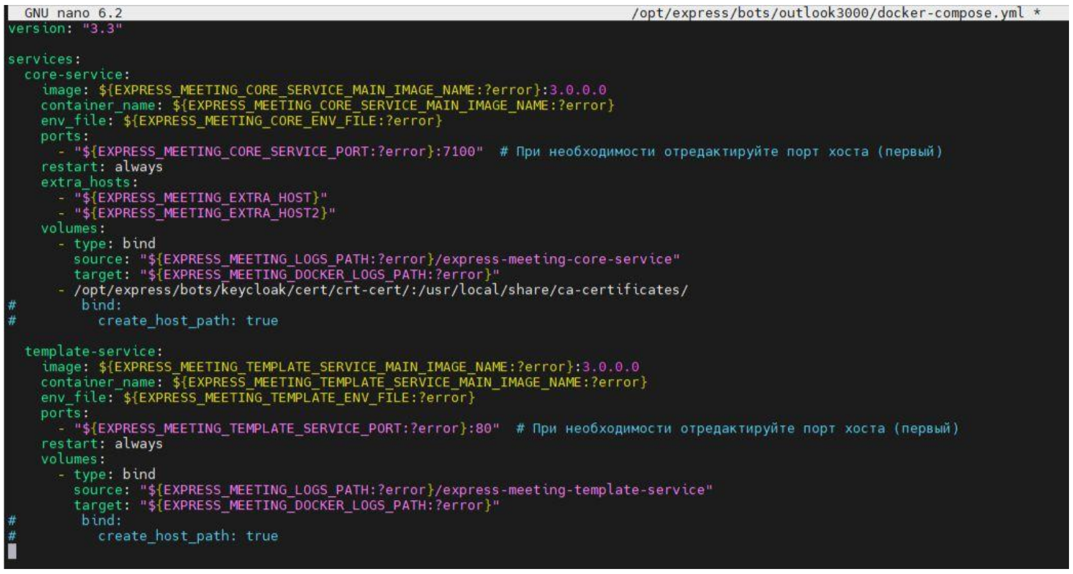
```
- /opt/express/bots/cert/:/usr/local/share/ca-certificates/
```

or

```
- /etc/ssl/certs/ca-certificates.crt:/etc/ssl/certs/ca-
certificates.crt
```

(if the SSL connection is established correctly from the Docker host).

Important! The first part of the string, /opt/express/bots/cert/, can be anything and must correspond to the path from step 2. The second part of this string, :/usr/local/share/ca-certificates/, is constant and must not be changed. The express-meeting-core-service-main takes all necessary certificates from the specified folder.



```
  GNU nano 6.2                                                        /opt/express/bots/outlook3000/docker-compose.yml *
version: "3.3"

services:
  core-service:
    image: ${EXPRESS_MEETING_CORE_SERVICE_MAIN_IMAGE_NAME:?error}:3.0.0.0
    container_name: ${EXPRESS_MEETING_CORE_SERVICE_MAIN_IMAGE_NAME:?error}
    env_file: ${EXPRESS_MEETING_CORE_ENV_FILE:?error}
    ports:
      - "${EXPRESS_MEETING_CORE_SERVICE_PORT:?error}:7100"  # При необходимости отредактируйте порт хоста (первый)
    restart: always
    extra_hosts:
      - "${EXPRESS_MEETING_EXTRA_HOST}"
      - "${EXPRESS_MEETING_EXTRA_HOST2}"
    volumes:
      - type: bind
        source: "${EXPRESS_MEETING_LOGS_PATH:?error}/express-meeting-core-service"
        target: "${EXPRESS_MEETING_DOCKER_LOGS_PATH:?error}"
      - /opt/express/bots/keycloak/cert/crt-cert/:/usr/local/share/ca-certificates/
#        bind:
#          create_host_path: true

  template-service:
    image: ${EXPRESS_MEETING_TEMPLATE_SERVICE_MAIN_IMAGE_NAME:?error}:3.0.0.0
    container_name: ${EXPRESS_MEETING_TEMPLATE_SERVICE_MAIN_IMAGE_NAME:?error}
    env_file: ${EXPRESS_MEETING_TEMPLATE_ENV_FILE:?error}
    ports:
      - "${EXPRESS_MEETING_TEMPLATE_SERVICE_PORT:?error}:80"  # При необходимости отредактируйте порт хоста (первый)
    restart: always
    volumes:
      - type: bind
        source: "${EXPRESS_MEETING_LOGS_PATH:?error}/express-meeting-template-service"
        target: "${EXPRESS_MEETING_DOCKER_LOGS_PATH:?error}"
#        bind:
#          create_host_path: true
```

Figure 21

6. Stop the Outlook add-in containers by running the following command from the folder with the configuration files:

   ```
   docker-compose down
   ```

7. Start the docker containers from the folder:

   ```
   docker-compose --env-file ./.env -f ./docker-compose.yml up -d
   ```

8. Access the express-meeting-core-service-main container using the command:

   ```
   docker exec -it express-meeting-core-service-main bash
   ```

9. Update the list of certificates with the command:

   ```
   update-ca-certificates
   ```

10. Exit the express-meeting-core-service-main container using the command:

    ```
    exit
    ```

11. Try creating a conference using the client software.
12. Check the system for the absence of SSL-related errors using the command:

    ```
    docker logs --tail=200 -f  express-meeting-core-service-main
    ```

Appendix 2

NETWORK INTERACTION

Table 4. Single CTS or Split CTS

No.	Source	Destination	Port and Protocol	Interaction Description
1.	Internal User (Client PC)	express-core-service (docker Host)	7100\TCP	Interaction of the Outlook add-in on the user's PC on CTS01 with express-core-service001 on the docker Host server (standard network port specified)
2.	express-core-service (docker Host)	CTS Front (CTS Single)	443\TCP	Interaction of express-core-service on docker Host with API Conference Notifier Bot on CTS Front (CTS Single) server
3.	Template Administrator (Client PC)	express-template-service (docker Host)	7200\TCP	Interaction of the Template Administrator with express-template-service001 on docker Host via the web interface (standard network port specified)
4.	express-core-service (docker Host)	PostgreSQL DB (DB server)	5432\TCP	Interaction of express-core-service001 on docker Host with PostgreSQL DB on the DB server (standard network port specified)
5.	express-template-service (docker Host)	PostgreSQL DB (DB server)	5432\TCP	Interaction of express-template-service on docker Host with PostgreSQL DB on the DB server (standard network port specified)
6.	Internal User (Client PC)	ADFS Functionality (docker Host)	8080\TCP	Interaction of the Outlook add-in on the user's PC on CTS01 with ADFS on the ADFS server (standard network port specified)
7.	express-core-service (docker Host)	ADFS Functionality (docker Host)	8080\TCP	Interaction of express-core-service001 on docker Host with ADFS on the ADFS server (standard network port specified)

Note. The table specifies standard network ports for interaction between client PCs, express-core-service, and ADFS (SSL is configured separately and is not considered in this table).

Table 5. Multiple CTS without ETS

No.	Source	Destination	Port and Protocol	Interaction Description
1.	Internal User on CTS01 (Client PC)	express-core-service001 (docker Host)	7100\TCP	Interaction of the Outlook add-in on the user's PC on CTS01 with express-core-service001 on docker Host (standard network port specified)
2.	Internal User on CTS02 (Client PC)	express-core-service002 (docker Host)	7300\TCP	Interaction of the Outlook add-in on the user's PC on CTS02 with express-core-service002 on docker Host (standard network port specified)
3.	Template Administrator (Client PC)	express-template-service (docker Host)	7200\TCP	Interaction of the Template Administrator with express-template-service on docker Host via the web interface (standard network port specified)
4.	express-core-service001 (docker Host)	PostgreSQL DB (DB server)	5432\TCP	Interaction of express-core-service001 on docker Host with PostgreSQL DB on the DB server (standard network port specified)
5.	express-template-service (docker Host)	PostgreSQL DB (DB server)	5432\TCP	Interaction of express-template-service on docker Host with PostgreSQL DB on the DB server (standard network port specified)
6.	express-core-service002 (docker Host)	PostgreSQL DB (DB server)	5432\TCP	Interaction of express-core-service002 on docker Host with PostgreSQL DB on the DB server (standard network port specified)
7.	express-core-service001 (docker Host)	CTS Front (CTS Single)01	443\TCP	Interaction of express-core-service001 on docker Host with API Conference Notifier Bot on CTS Front (CTS Single)01 server
8.	express-core-service002 (docker Host)	CTS Front (CTS Single)02	443\TCP	Interaction of express-core-service002 on docker Host with API Conference Notifier Bot on CTS02 Front (CTS02 Single)02 server
9.	express-core-service001 (docker Host)	ADFS Functionality (docker Host)	8080\TCP	Interaction of express-core-service001 on docker Host with ADFS on the ADFS server (standard network port specified)
10.	Internal User (Client PC)	ADFS Functionality (docker Host)	8080\TCP	Interaction of the Outlook add-in on the user's PC on CTS01 with ADFS on the ADFS server (standard network port specified)
11.	express-core-service002 (docker Host)	ADFS Functionality (docker Host)	8080\TCP	Interaction of express-core-service002 on docker Host with ADFS on the ADFS server (standard network port specified)
12.	Internal User (Client PC)	ADFS Functionality (docker Host)	8080\TCP	Interaction of the Outlook add-in on the user's PC on CTS02 with ADFS on the ADFS server (standard network port specified)

Note. The table specifies standard network ports for interaction between client PCs, express-core-service, and ADFS (SSL is configured separately and is not considered in this table).

Table 6. Multiple CTS with ETS

No.	Source	Destination	Port and Protocol	Interaction Description
1.	Internal User on CTS1 (Client PC)	express-core-service (docker Host)	7100\TCP	Interaction of the Outlook add-in on the user's PC on CTS01 with express-core-service001 on docker Host (standard network port specified)
2.	Internal User on CTS2 (Client PC)	express-core-service (docker Host)	7100\TCP	Interaction of the Outlook add-in on the user's PC on CTS02 with express-core-service001 on docker Host (standard network port specified)
3.	Template Administrator (Client PC)	express-template-service (docker Host)	7200\TCP	Interaction of the Template Administrator with express-template-service on docker Host via the web interface (standard network port specified)
4.	express-core-service001 (docker Host)	PostgreSQL DB (DB server)	5432\TCP	Interaction of express-core-service001 on docker Host with PostgreSQL DB on the DB server (standard network port specified)
5.	express-template-service (docker Host)	PostgreSQL DB (DB server)	5432\TCP	Interaction of express-template-service on docker Host with PostgreSQL DB on the DB server (standard network port specified)
6.	express-core-service (docker Host)	ETS EXPRESS Server	443\TCP	Interaction of express-core-service on docker Host with API Conference Notifier Bot on the ETS EXPRESS Server
7.	express-core-service (Docker Host)	ETS EXPRESS Server	443\TCP	Interaction of express-core-service001 on docker Host with API Conference Notifier Bot on the ETS EXPRESS Server
8.	express-core-service001 (docker Host)	ADFS Functionality (docker Host)	8080\TCP	Interaction of express-core-service001 on docker Host with ADFS on the ADFS server (standard network port specified)
9.	Internal User (Client PC)	ADFS Functionality (docker Host)	8080\TCP	Interaction of the Outlook add-in on the user's PC on CTS01 with ADFS on the ADFS server (standard network port specified)
10.	Internal User (Client PC)	ADFS Functionality (docker Host)	8080\TCP	Interaction of the Outlook add-in on the user's PC on CTS02 with ADFS on the ADFS server (standard network port specified)

Note:

- The table specifies standard network ports for interaction between client PCs, express-core-service, and ADFS (SSL is configured separately and is not considered in this table).
- Network interaction between EXPRESS ETS servers and EXPRESS CTS servers is described in detail in “Express. Communication System. Administrator's Guide. Volume 1. Installation” (https://express.ms/admin_guide_install.pdf).