CRMLXRMS Labs

Amadeusz Nowak, Andrii Voznesenskyi, Piotr Padamczyk

Aug 09, 2024

CONTENTS:

Add your content using `reStructuredText` syntax. See the reStructuredText documentation for details.

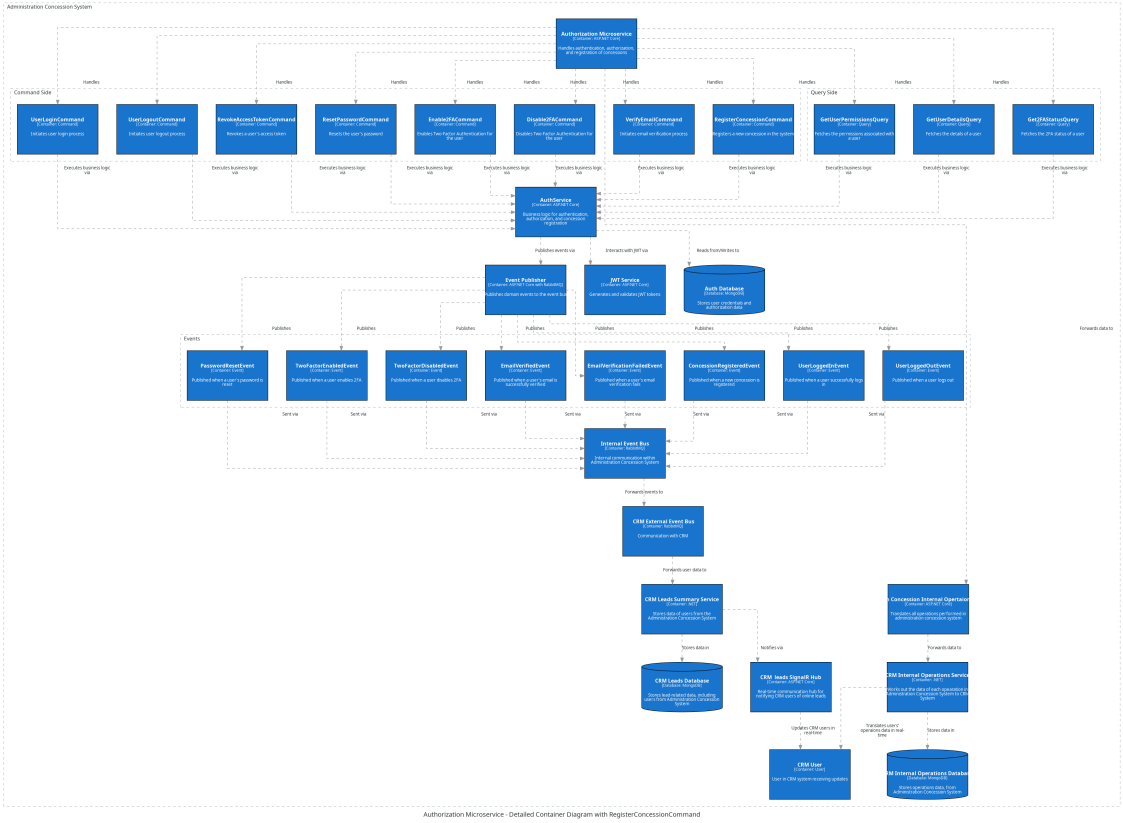
CHAPTER

ONE

ADMINISTRATION CONCESSION SYSTEM

1.1 Administration Concession System Authorization Service

The Authorization Microservice is a core component of the Administration Concession System. It handles authentication, authorization, and user management processes, which are essential for maintaining secure access and operational control. This microservice also integrates with the CRM system to share user data and operational events.



1.1.1 API Documentation

1.1.2 Commands

UserLoginCommand

- **Description**: Initiates the process of logging a user into the system.
- **Request URL**: */api/auth/login*
- **Method**: *POST*
- **Request Body**: - *username* (string, required): The username of the user. - *password* (string, required): The password of the user.
- **Response**: - *token* (string): A JWT token for authenticated sessions. - *expiresIn* (int): Expiration time of the token in seconds.
- **Events Triggered**: *UserLoggedInEvent*

UserLogoutCommand

- **Description**: Initiates the process of logging a user out of the system.
- **Request URL**: */api/auth/logout*
- **Method**: *POST*
- **Request Body**: None
- **Response**: - *message* (string): Confirmation message that the user has been logged out.
- **Events Triggered**: *UserLoggedOutEvent*

RevokeAccessTokenCommand

- **Description**: Revokes a user's access token, invalidating their current session.
- **Request URL**: */api/auth/revoke-token*
- **Method**: *POST*
- **Request Body**: - *token* (string, required): The access token to be revoked.
- **Response**: - *message* (string): Confirmation message that the token has been revoked.
- **Events Triggered**: *UserLoggedOutEvent*

ResetPasswordCommand

- **Description**: Resets the password for a user's account.
- **Request URL**: */api/auth/reset-password*
- **Method**: *POST*
- **Request Body**: - *email* (string, required): The email associated with the user's account.
- **Response**: - *message* (string): Confirmation that a password reset email has been sent.
- **Events Triggered**: *PasswordResetEvent*

Enable2FACommand

- **Description**: Enables Two-Factor Authentication (2FA) for a user's account.
- **Request URL**: */api/auth/enable-2fa*
- **Method**: *POST*

- **Request Body**: - *userId* (string, required): The ID of the user enabling 2FA. - *token* (string, required): The 2FA token generated by the user's authenticator app.
- **Response**: - *message* (string): Confirmation that 2FA has been enabled.
- **Events Triggered**: *TwoFactorEnabledEvent*

Disable2FACommand

- **Description**: Disables Two-Factor Authentication (2FA) for a user's account.
- **Request URL**: */api/auth/disable-2fa*
- **Method**: *POST*
- **Request Body**: - *userId* (string, required): The ID of the user disabling 2FA.
- **Response**: - *message* (string): Confirmation that 2FA has been disabled.
- **Events Triggered**: *TwoFactorDisabledEvent*

VerifyEmailCommand

- **Description**: Initiates the process of verifying a user's email address.
- **Request URL**: */api/auth/verify-email*
- **Method**: *POST*
- **Request Body**: - *userId* (string, required): The ID of the user verifying their email. - *emailToken* (string, required): The token sent to the user's email.
- **Response**: - *message* (string): Confirmation that the email has been verified.
- **Events Triggered**: *EmailVerifiedEvent*, *EmailVerificationFailedEvent*

RegisterConcessionCommand

- **Description**: Registers a new concession within the system.
- **Request URL**: */api/auth/register-concession*
- **Method**: *POST*
- **Request Body**: - *concessionName* (string, required): The name of the concession. - *adminEmail* (string, required): The email of the concession's administrator.
- **Response**: - *message* (string): 200 – Confirmation that the concession has been registered.
- **Events Triggered**: *ConcessionRegisteredEvent*

1.1.3 Queries

GetUserPermissionsQuery

- **Description**: Retrieves the permissions associated with a user.
- **Request URL**: */api/auth/get-user-permissions*
- **Method**: *GET*
- **Request Parameters**: - *userId* (string, required): The ID of the user whose permissions are being retrieved.
- **Response**: - *permissions* (array of strings): A list of permissions associated with the user.

GetUserDetailsQuery

- **Description**: Fetches the details of a specific user.

- **Request URL**: */api/auth/get-user-details*
- **Method**: *GET*
- **Request Parameters**: - *userId* (string, required): The ID of the user whose details are being fetched.
- **Response**: - *userId* (string): The ID of the user. - *username* (string): The username of the user. - *email* (string): The email of the user. - *roles* (array of strings): A list of roles assigned to the user.

Get2FAStatusQuery

- **Description**: Retrieves the 2FA (Two-Factor Authentication) status of a user.
- **Request URL**: */api/auth/get-2fa-status*
- **Method**: *GET*
- **Request Parameters**: - *userId* (string, required): The ID of the user whose 2FA status is being retrieved.
- **Response**: - *is2FAEnabled* (boolean): Indicates whether 2FA is enabled for the user.

1.1.4 Inbox Events

The Authorization Microservice processes the following inbox events:

ChangeUserRightsEvent

- **Description**: An event that triggers when a user's rights are changed within the CRM system.
- **Source**: CRM System – Leads summary/managenent service (External CRM event bus brigged to Administration concession system).
- **Actions**: Updates the user's permissions within the Administration Concession System to reflect the changes made in the CRM system.

BlockUserAccessEvent

- **Description**: An event that triggers when a user is blocked or suspended in the CRM system.
- **Source**: CRM System – Leads summary/managenent service (External CRM event bus brigged to Administration concession system).
- **Actions**: Immediately revokes the user's access to the Administration Concession System, ensuring that the block is enforced across systems.

UnBlockUserAccessEvent

- **Description**: An event that triggers when a user is unblocked in case of being suspended in the CRM system.
- **Source**: CRM System – Leads summary/managenent service (External CRM event bus brigged to Administration concession system).
- **Actions**: Immediately revokes the user's access to the Administration Concession System, ensuring that the block is enforced across systems.

1.1.5 Outbox Events

The Authorization Microservice emits the following outbox events:

UserLoggedInEvent

- **Description**: Emitted after a successful user login within the Administration Concession System.
- **Destination**: Administration Concession system Internal operation serivce, CRM System Leads Summary Service (leads hub connector service), CRM GRPC intergnal operations service.
- **Actions**: Notifies the Administration Concession Administrator user and CRM system user of the user's login, updates session details.

User2FaAccessTokenGenerated

- **Description**: Emitted after a successful user login within the password and login to the Administration Concession System but having the 2fa enabled.
- **Destination**: Administration Concession system Notification service, Administration Concession System Internal Operations service.
- **Actions**: Required Noitification service Event Emition to pass the token generated back to User trying to sign-in, Notify Administration of Concession system of having the user trying to access the system with 2fa token.

UserLoggedOutEvent

- **Description**: Emitted after a user logs out within the Administration Concession System.
- **Destination**: Administration Concession system Accouns summary service, Administration Concession System Internal Operations Service, CRM Internal Operaions service.
- **Actions**: Notifies the CRM system that the user has logged out (updated leads connection hub), updates session termination details.

PasswordResetEvent

- **Description**: Emitted after a user's password is reset within the Administration Concession System.
- **Destination**: CRM System Operation service, CRM leads summary service, Administration Concession system Accouns summary service, Administration Concession System Internal Operations Service, CRM Internal Operaions service.
- **Actions**: Notifies the Administration Concession System administrator, CRM system of the specifi user (lead in the case of CRM) password reset, triggers notifications in Administration Concession System Notifcation service (which will be possibly passed to Email Service), and updates security logs.

TwoFactorEnabledEvent

- **Description**: Emitted after Two-Factor Authentication is enabled for a user within the Administration Concession System.
- **Destination**: CRM System Operation service, CRM leads summary service, Administration Concession system Accouns summary service, Administration Concession System Internal Operations Service, CRM Internal Operaions service. .
- **Actions**: Notifies the Administration Concession System administrator, CRM system of the specifi user update, updates user security settings, and ensures that 2FA is enforced consistently across AConS.

TwoFactorDisabledEvent

- **Description**: Emitted after Two-Factor Authentication is disabled for a user within the Administration Concession System.

- **Destination**: CRM System Operation service, CRM leads summary service, Administration Concession system Accouns summary service, Administration Concession System Internal Operations Service, CRM Internal Operaions service. .
- **Actions**: Notifies the Administration Concession System administrator, CRM system of the specifi user update, updates user security settings, and ensures that 2FA is enforced consistently across AConS.

EmailVerifiedEvent

- **Description**: Emitted when a user's email is successfully verified within the Administration Concession System.
- **Destination**: CRM System Operation service, CRM leads summary service, Administration Concession system Accouns summary service, Administration Concession System Internal Operations Service, CRM Internal Operaions service.
- **Actions**: Notifies the Administration Concession System administrator, CRM system of specific user (lead) email verification, updates the user's profile.

EmailVerificationFailedEvent

- **Description**: Emitted when a user's email verification fails within the Administration Concession System.
- **Destination**: CRM System Operation service, CRM leads summary service, Administration Concession system Accouns summary service, Administration Concession System Internal Operations Service, CRM Internal Operaions service.
- **Actions**: Notifies the Administration Concession System administrator, CRM system of specific user (lead) failed verification, updates logs, and marks the user's email as unverified.

ConcessionRegisteredEvent

- **Description**: Emitted after a new concession is registered within the Administration Concession System.
- **Destination**: CRM System Orders service, Administration Concession system Accouns summary service, Administration Concession System Internal Operations Service, CRM Internal Operaions service.
- **Actions**: Notifies the Administration Concession System administrator, CRM system of the new concession, triggers onboarding processes, and syncs concession details across systems.

1.2 Authorization Service Database Documentation

The *Authorization Service* manages the storage and retrieval of authentication, authorization, and user management data within the Administration Concession System. The database contains several key documents that are essential for handling user identities, permissions, and authentication mechanisms.

1.2.1 UserDocument

Represents the core information related to a user in the system. This document is essential for managing user authentication, authorization, and profile details.

- **Id**: *Guid* - Unique identifier for the user.
- **Username**: *string* - The username chosen by the user.
- **Email**: *string* - The email address associated with the user's account.
- **Role**: *string* - The role assigned to the user (e.g., *Admin*, *User*).
- **PasswordHash**: *string* - The hashed password for the user's account.
- **CreatedAt**: *DateTime* - The timestamp when the user account was created.

- **IsEmailVerified**: *bool* - Indicates whether the user's email has been verified.
- **EmailVerificationToken**: *string* - The token used for verifying the user's email address.
- **EmailVerifiedAt**: *DateTime?* - The timestamp when the user's email was verified.
- **IsTwoFactorEnabled**: *bool* - Indicates whether Two-Factor Authentication (2FA) is enabled for the user.
- **TwoFactorSecret**: *string* - The secret key used for generating 2FA codes.

1.2.2 RefreshTokenDocument

Stores the refresh tokens associated with a user's session. These tokens are used to renew access tokens without requiring the user to re-authenticate.

- **Id**: *Guid* - Unique identifier for the refresh token.
- **UserId**: *Guid* - The identifier of the user to whom the refresh token belongs.
- **Token**: *string* - The actual refresh token value.
- **CreatedAt**: *DateTime* - The timestamp when the refresh token was created.
- **RevokedAt**: *DateTime?* - The timestamp when the refresh token was revoked (if applicable).

1.2.3 UserResetTokenDocument

Manages the tokens used for resetting a user's password. This document is crucial for handling the password reset flow securely.

- **Id**: *Guid* - Unique identifier for the reset token.
- **UserId**: *Guid* - The identifier of the user requesting the password reset.
- **ResetToken**: *string* - The token sent to the user for resetting their password.
- **ResetTokenExpires**: *DateTime?* - The expiration timestamp for the reset token.

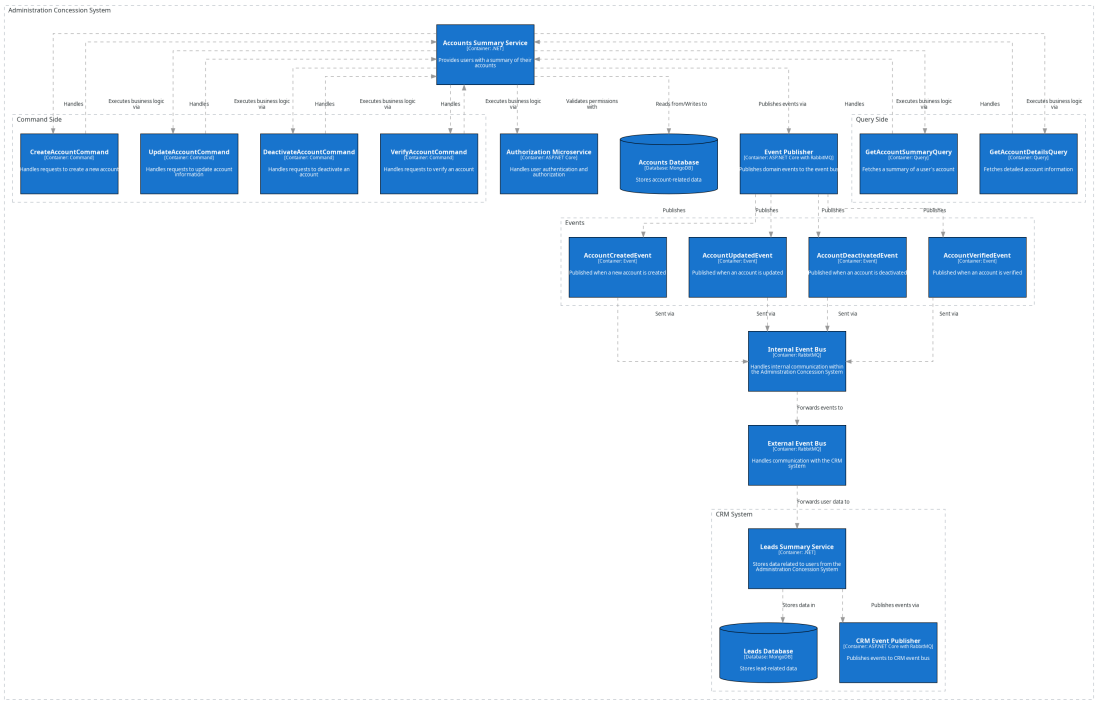
1.2.4 PermissionsDocument

Represents the permissions that a user has within the system. This document is crucial for determining the specific actions a user is allowed to perform based on the roles and permissions assigned to them.

- **Id**: *Guid* - Unique identifier for the permission entry.
- **UserId**: *Guid* - The identifier of the user to whom the permissions belong.
- **Permissions**: *IEnumerable<string>* - A list of permission strings that define the actions the user is allowed to perform (e.g., *ViewDashboard*, *EditConcession*, *ManageUsers*).
- **AssignedAt**: *DateTime* - The timestamp when these permissions were assigned to the user.
- **ExpiresAt**: *DateTime?* - The optional timestamp indicating when these permissions expire (if applicable).

1.3 Administration Concession System Accounts Summary Service

The Accounts Summary Service is an integral component of the Administration Concession System, designed to manage and provide a summary of administration concession user accounts. This service facilitates various operations related to accounts, including creation, updates, permissions updates, deactivation, and verification. It also supports queries for retrieving account summaries and detailed account information. The permission synchronization will hapeent through the permission service: after user with the corresponding permission will create the pecific user role with the specific permission assignt the permissions set and the role to the specific user the corresponding repository in the studnet service will be udpated do to inbox event from permission service. This optimisation is required for the not biased http request. However, in this case, the one more problem appears: for the moment the service with permissions is not available there will not be possible to provide any permission update for the specific role. For this time to situation when the new user has been created accured after the Permission Service will be available there should be a check available and additional outbox evnet publiching for such sort of situations. Theoretically, also thie situation should be in qute informative way described in the Permission service of the Administration Concesstion system.



Accounts Summary Service - Complex Container Diagram

1.3.1 API Documentation

1.3.2 Commands

CreateAccountCommand

- **Description**: Handles the creation of a new user account within the system.
- **Request URL**: */api/accounts/create*
- **Method**: *POST*
- **Request Body**: - *userId* (string, required): Unique identifier for the user. - *accountDetails* (object, required): Detailed information about the account (e.g., email, name, etc.).
- **Response**: - *accountId* (string): The ID of the newly created account. - *message* (string): Confirmation that the account has been created.
- **Events Triggered**: *AccountCreatedEvent*

UpdateAccountCommand

- **Description**: Handles updates to an existing user account's information.
- **Request URL**: */api/accounts/update*
- **Method**: *PUT*
- **Request Body**: - *accountId* (string, required): The ID of the account to be updated. - *updatedDetails* (object, required): Updated account information (e.g., new email, name changes, etc.).
- **Response**: - *message* (string): Confirmation that the account has been updated.
- **Events Triggered**: *AccountUpdatedEvent*

DeactivateAccountCommand

- **Description**: Handles the deactivation of an existing user account.
- **Request URL**: */api/accounts/deactivate*
- **Method**: *POST*
- **Request Body**: - *accountId* (string, required): The ID of the account to be deactivated. - *reason* (string, optional): Reason for deactivation.
- **Response**: - *message* (string): Confirmation that the account has been deactivated.
- **Events Triggered**: *AccountDeactivatedEvent*

VerifyAccountCommand

- **Description**: Handles the verification process of a user account.
- **Request URL**: */api/accounts/verify*
- **Method**: *POST*
- **Request Body**: - *accountId* (string, required): The ID of the account to be verified. - *verificationDetails* (object, required): Details required for verification (e.g., verification code).
- **Response**: - *message* (string): Confirmation that the account has been verified.
- **Events Triggered**: *AccountVerifiedEvent*

1.3.3 Queries

GetAccountSummaryQuery

- **Description**: Retrieves a summary of a user account.
- **Request URL**: */api/accounts/summary*
- **Method**: *GET*
- **Request Parameters**: - *accountId* (string, required): The ID of the account for which the summary is requested.
- **Response**: - *accountId* (string): The ID of the account. - *summary* (object): A brief summary of the account, including key details (e.g., account status, creation date).
- **Events Triggered**: None

GetAccountDetailsQuery

- **Description**: Fetches detailed information about a user account.
- **Request URL**: */api/accounts/details*
- **Method**: *GET*
- **Request Parameters**: - *accountId* (string, required): The ID of the account for which details are requested.
- **Response**: - *accountId* (string): The ID of the account. - *details* (object): Detailed information about the account, including all associated data (e.g., user profile, activity log).
- **Events Triggered**: None

1.3.4 Inbox Events

The Accounts Summary Service processes the following inbox events:

AccountCreatedEvent

- **Description**: Emitted after a new account is successfully created.
- **Source**: Administration Concession Authorization Service
- **Actions**: Notifies the system that a new account has been created, triggering further actions in other services (e.g., Concession Onboarding).

UserLoggedOutEvent

- **Description**: An event that triggers when a user logs out of the system.
- **Source**: Authorization Service
- **Actions**: Updates the account summary to reflect the user's last logout time.

PasswordResetEvent

- **Description**: An event that triggers when a user's password is reset.
- **Source**: Administration Concession Authorization Service
- **Actions**: Updates the account summary with information about the password reset, including the timestamp and fingerprints any relevant security notes.

TwoFactorAuthenticationMethodEnabledEvent

- **Description**: An event that triggers when a users sets his 2fa to be anabled.
- **Source**: Administration Concession Authorization Service

- **Actions**: Updates the account summary with information about the 2fa method enabled for the user permission verification, including the timestamps and fingerprints. Also, thie event mey be believed to be redundant, as the infomration about the 2fa verificaition will also be available in the Claims generated from token.

UserAssignedToPermissionsSetEvent/UserAssignedToRoleEvent

- **Description**: And event from the Permission service in the case any manipulation with the user permission service happens.
- **Source**: Administration Concession Permission Service
- **Actions**: Updates the account summary with information about the specific set of rht user's permissions, including the timestamp and any relevant security notes. This is crutial for the query of Getting the User Entity with the speicific set of permission user have assigned, which will be updated be the message recieved from Permissions Service. to provide an optimistaion level for the Accounts Summary service in administration Concession system.

1.3.5 Outbox Events

The Accounts Summary Service emits the following outbox events:

AccountUpdatedEvent

- **Description**: Emitted after an account is successfully updated in comparisson to the previous account data.
- **Destination**: Administration Concession Event Bus, CRM event Bus, CRM System Leads summary service
- **Actions**: Notifies the system of the account update, ensuring all related services have the most current account information.

AccountDeactivatedEvent

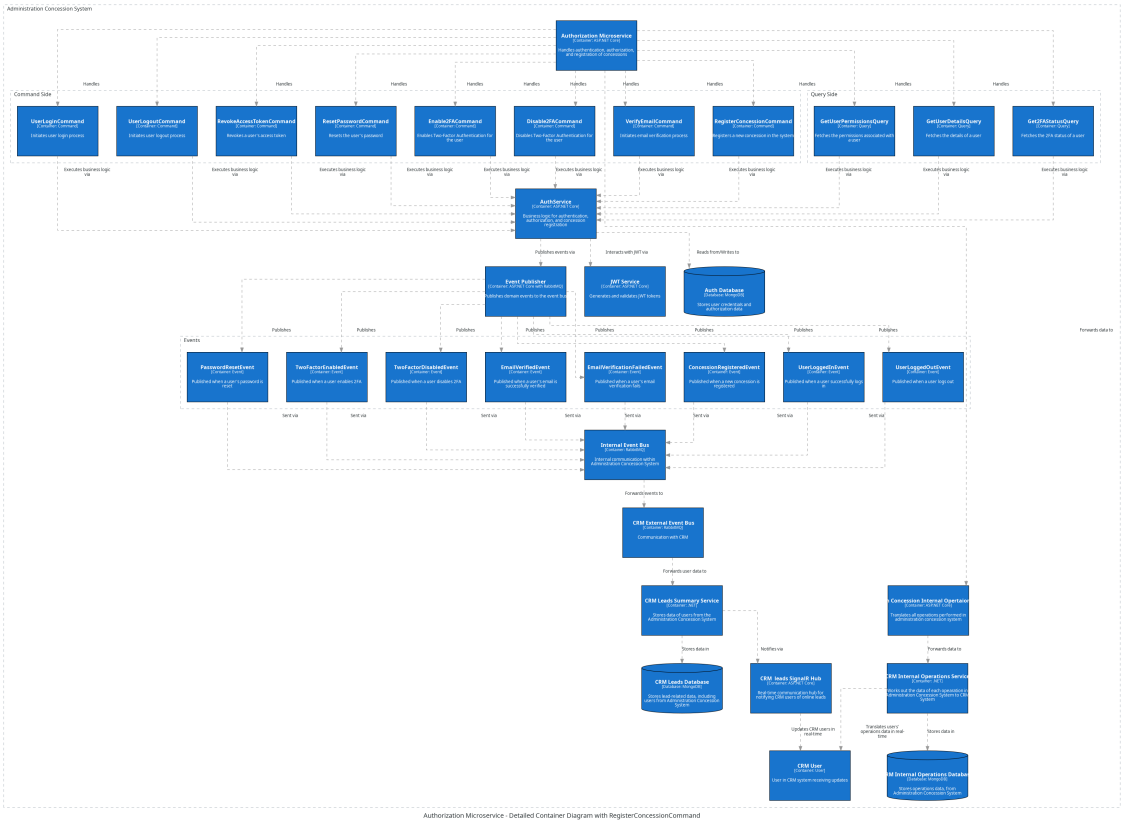
- **Description**: Emitted after an account is deactivated.
- **Destination**: Administration Concession Event Bus, CRM System Leads summary service
- **Actions**: Notifies the system and associated services that the account has been deactivated, triggering any necessary follow-up actions.

AccountVerifiedEvent

- **Description**: Emitted after an account is verified.
- **Destination**: Administration Concession Event Bus, CRM System
- **Actions**: Notifies the Concessions System and CRM that the account has been verified, updating the status across all relevant services.

1.4 Administration Concession System Internal Statistics Service

The Intrnal Statistics Microservice is responsible for handling recieving and saving the CRM Intrnal Statistics Service data: services distribution, situation characteristics, current leads summary tendency, general leads tandency



1.4.1 API Documentation

1.4.2 Commands

1.4.3 Queries

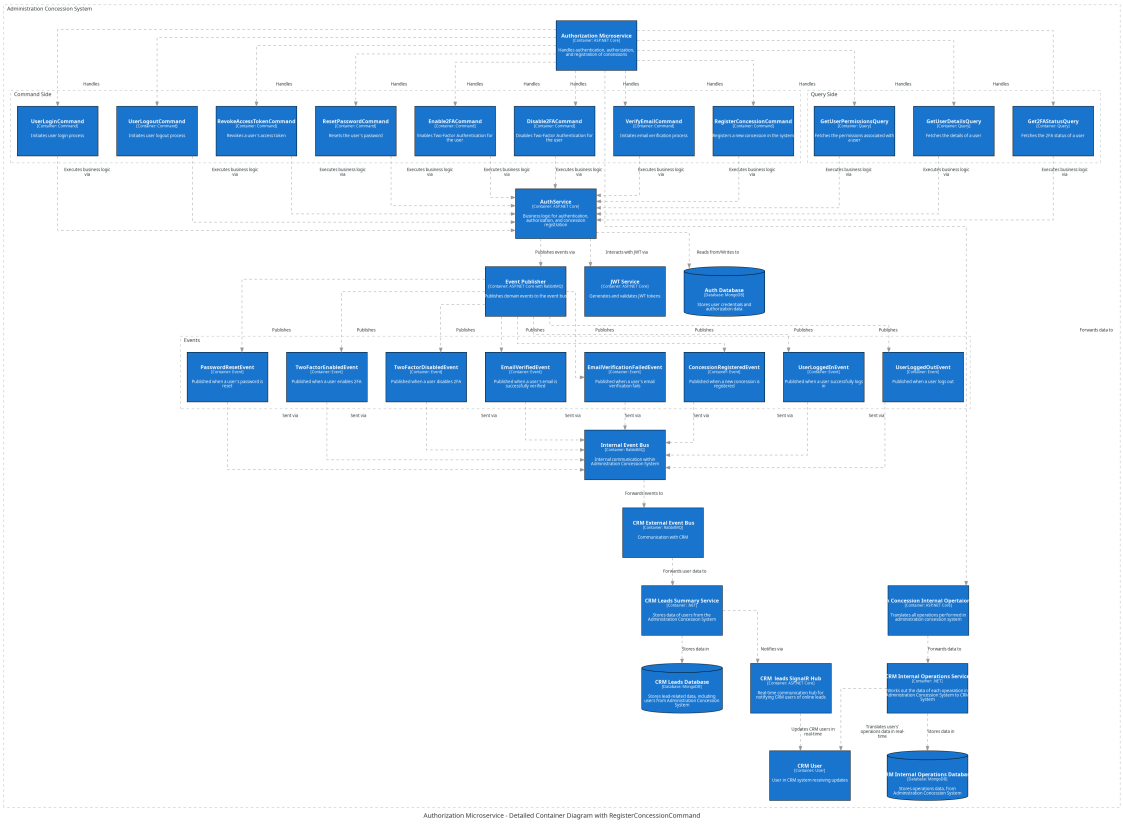
1.4.4 Inbox Events

1.4.5 Outbox Events

1.5 Internal Statistics Service Database Documentation

1.6 Administration Concession System External Statistics Service

The External Statistics Microservice is responsible for handling recieving From External Statistics service Of CRM connected to the shole Data procession Mining, Procession and analysis Machine, concession related statistics with regression analysis of behaviour, predictions. hypothesis verification, decission for the specific clinet's concessionices.



1.6.1 API Documentation

1.6.2 Commands

1.6.3 Queries

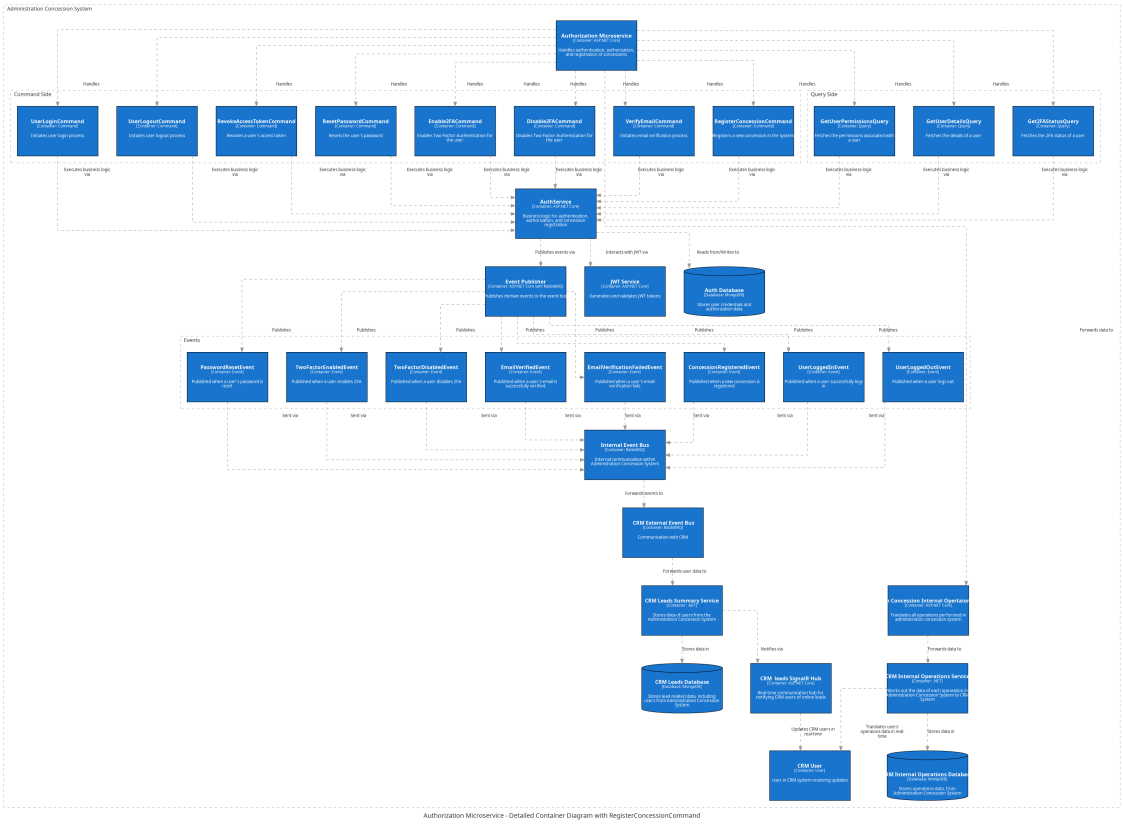
1.6.4 Inbox Events

1.6.5 Outbox Events

1.7 External Statistics Service Database Documentation

1.8 Administration Concession System Payments Service

The Payments Microservice is responsible for handing the data related to payments for the specific ordes for the leads recieved from the CRM Payments service (CRM payments service will be connected to the CFM Orders servcie and the CRM Facturation service and the CRM Leads Summary servcie)



1.8.1 API Documentation

1.8.2 Commands

1.8.3 Queries

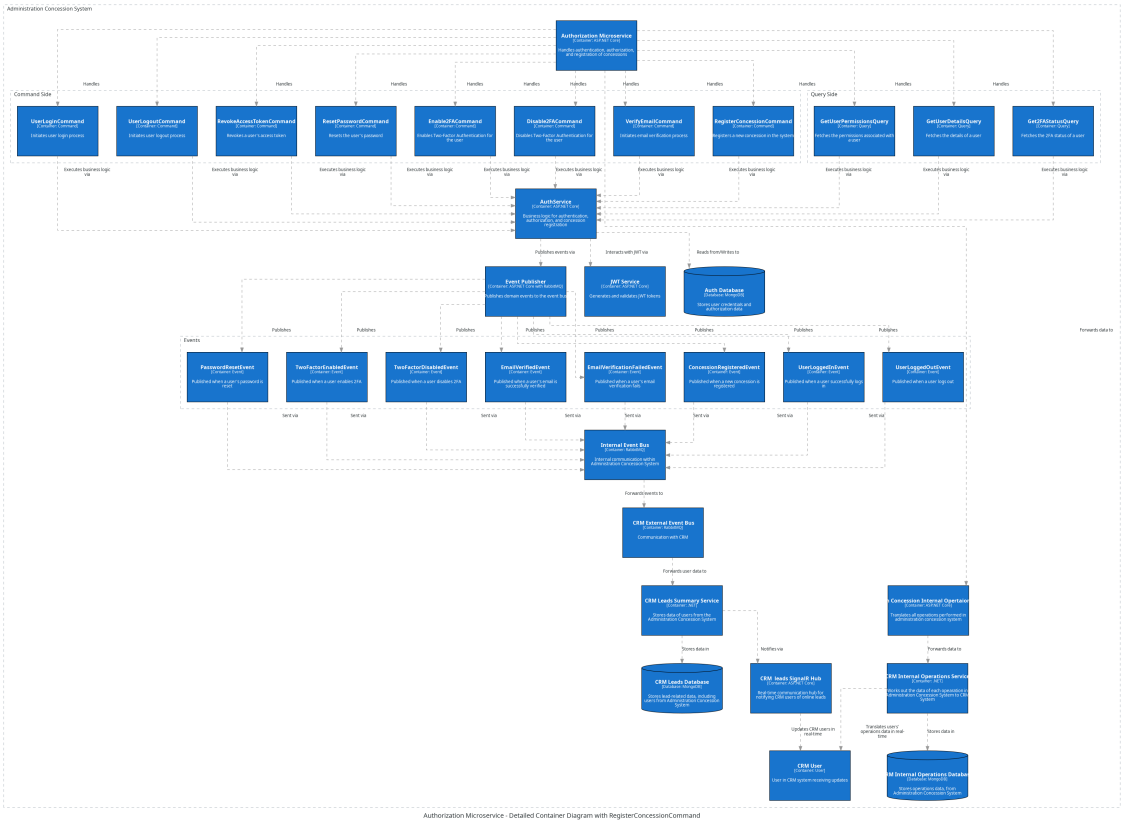
1.8.4 Inbox Events

1.8.5 Outbox Events

1.9 Payments Service Database Documentation

1.10 Administration Concession System Permissions Service

The Permission Microservice is responsible for handing the any actions with the permission related issues: creation the roles (out of permission set), defining the element of permissions set with the correspongin accesses. Thie Servcie will be also integrated with the CRM Leads summary servcie and CRM Permission service as theoretically the permisision assigned to the manager of the concesison system are also the permission of the userd from the crm assigned to serving Administration Concession System.



1.10.1 API Documentation

1.10.2 Commands

1.10.3 Queries

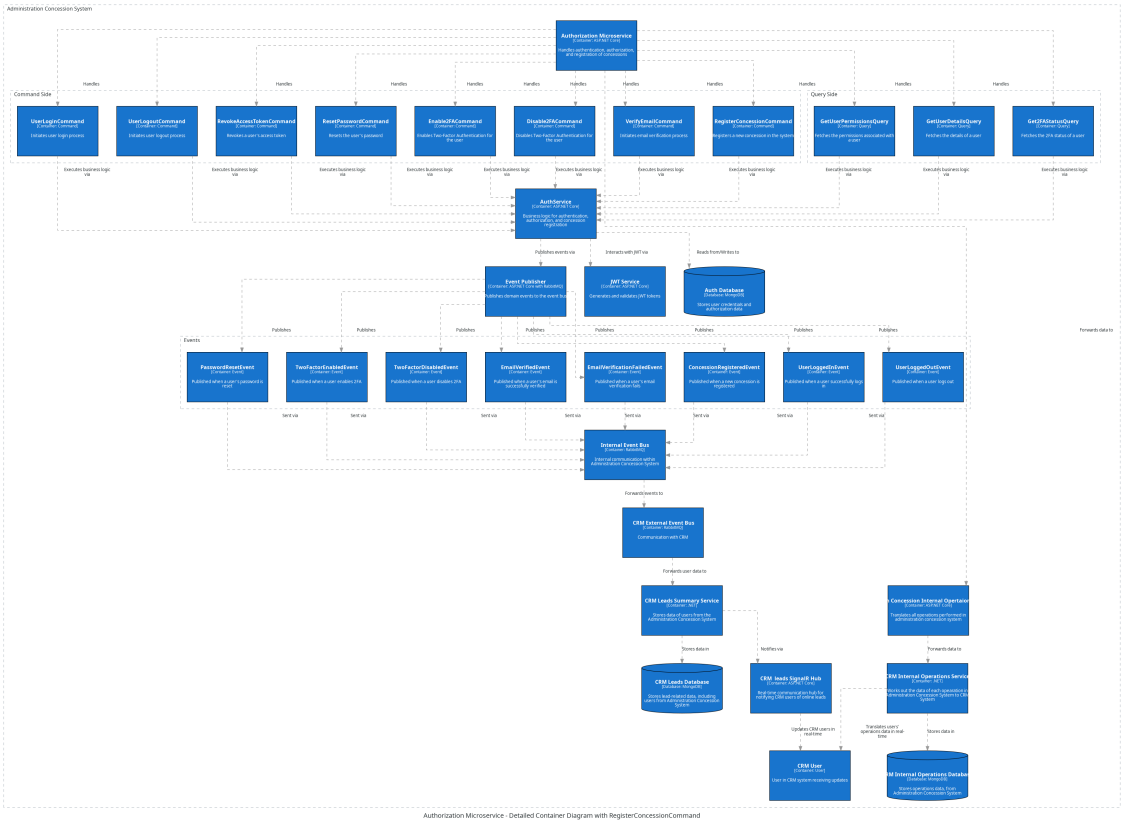
1.10.4 Inbox Events

1.10.5 Outbox Events

1.11 Permissions Service Database Documentation

1.12 Administration Concession System Tasks Summary Service

The Tasks Summary service is a sort of the PRO-User services. This service is responsible for handling the data reltated to the tasks set to users or assigned by users or tries to be organizes by users. Is synchonized with the CRM Task Summary Servcie.



Authorization Microservice - Detailed Container Diagram with RegisterConcessionCommand

1.12.1 API Documentation

1.12.2 Commands

1.12.3 Queries

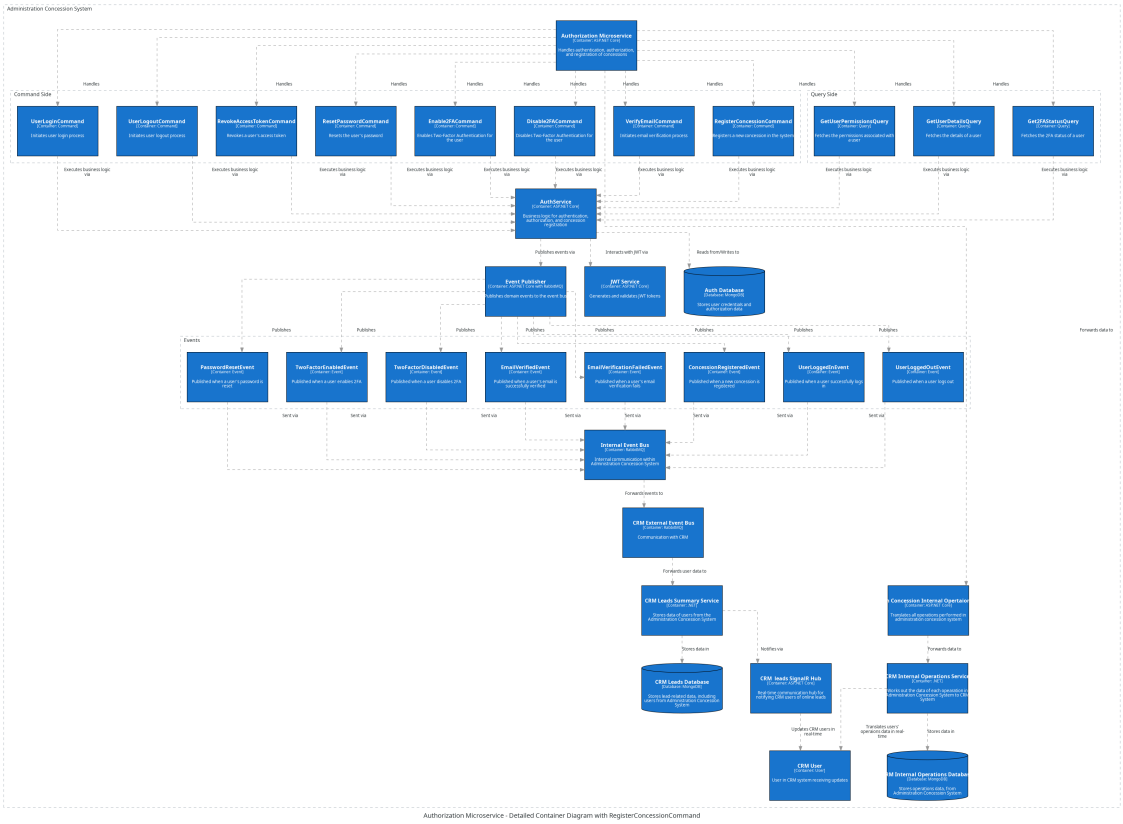
1.12.4 Inbox Events

1.12.5 Outbox Events

1.13 Tasks Summary Service Database Documentation

1.14 Administration Concession System Payments Service

The Orders Statistics Microservice is responsible for handling recieving the orders related data from the CRM Orders service.



1.14.1 API Documentation

1.14.2 Commands

1.14.3 Queries

1.14.4 Inbox Events

1.14.5 Outbox Events

1.15 Payments Service Database Documentation

CHAPTER

TWO

CRM SYSTEM