User Management Subsystem New

▼ public class User

1. Class Information
    - Class Name: User
    - Purpose: Represents a user entity
2. Attributes and Methods
    - Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique identifier for the user (UUID format).
localeId	private	String	Identifier for the user's locale.
roleId	private	String	Identifier for the user's role.
username	private	String	User's username.
password	private	String	User's password (hashed).
email	private	String	User's email address.
theme	private	String	User's selected theme preference.

displayTitleWeb	private	boolean	Whether to display only article titles in the web app.
displayTitleMobile	private	boolean	Whether to display only article titles in the mobile app.
displayUnreadWeb	private	boolean	Whether to display only unread articles in the web app.
displayUnreadMobile	private	boolean	Whether to display only unread articles in the mobile app.
narrowArticle	private	boolean	Whether to show narrow articles on a wide screen.
firstConnection	private	boolean	Whether the user has dismissed the first connection screen.
createDate	private	Date	Date when the user account was created.
deleteDate	private	Date	Date when the user account was deleted (null if active).

- Methods:

Method	Access Modifier	Description
String getId()	public	Returns the user ID.
void setId(String id)	public	Sets the user ID.

String getLocaleId()	public	Returns the locale ID.
void setLocaleId(String localeId)	public	Sets the locale ID.
String getRoleId()	public	Returns the role ID.
void setRoleId(String roleId)	public	Sets the role ID.
String getUsername()	public	Returns the username.
void setUsername(String username)	public	Sets the username.
String getPassword()	public	Returns the password.
void setPassword(String password)	public	Sets the password.
String getEmail()	public	Returns the email.
void setEmail(String email)	public	Sets the email.
String getTheme()	public	Returns the theme.
void setTheme(String theme)	public	Sets the theme.
boolean isDisplayTitleWeb()	public	Returns whether only article titles are displayed in the web app.
void setDisplayTitleWeb(boolean displayTitleWeb)	public	Sets whether only article titles are displayed in the web app.
boolean isDisplayTitleMobile()	public	Returns whether only article titles are displayed in the mobile app.
void setDisplayTitleMobile(boolean displayTitleMobile)	public	Sets whether only article titles are displayed in the mobile app.
boolean isDisplayUnreadWeb()	public	Returns whether only unread articles are displayed in the web app.

void setDisplayUnreadWeb(boolean displayUnreadWeb)	public	Sets whether only unread articles are displayed in the web app.
boolean isDisplayUnreadMobile()	public	Returns whether only unread articles are displayed in the mobile app.
void setDisplayUnreadMobile(boolean displayUnreadMobile)	public	Sets whether only unread articles are displayed in the mobile app.
boolean isNarrowArticle()	public	Returns whether narrow articles are shown on a wide screen.
void setNarrowArticle(boolean narrowArticle)	public	Sets whether narrow articles are shown on a wide screen.
boolean isFirstConnection()	public	Returns whether the first connection screen has been dismissed.
void setFirstConnection(boolean firstConnection)	public	Sets whether the first connection screen has been dismissed.
Date getCreateDate()	public	Returns the creation date of the user.
void setCreateDate(Date createDate)	public	Sets the creation date of the user.
Date getDeleteDate()	public	Returns the deletion date of the user.
void setDeleteDate(Date deleteDate)	public	Sets the deletion date of the user.

String toString()	public	Returns a string representation of the user object.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
        - Aggregation:
            - Role
            - Locale
        - Composition:

▼ public class UserDto

1. Class Information

- **Class Name:** `UserDto`
- **Purpose:** Represents user profile information, including metadata such as locale, email, and timestamps.

---

2. Attributes and Methods

Attributes

Attribute Name	Type	Description
`id`	`String`	Unique identifier for the user.
`localeId`	`String`	Locale identifier for language preferences.
`username`	`String`	Display name of the user.
`email`	`String`	Email address of the user.
`createTimestamp`	`Long`	Timestamp indicating user creation date.

Methods

- **Getters and Setters:**
  Standard
  `get` and `set` methods for each attribute.

---

3. Relationships : NA

▼ public class UserDao

---

1. Class Information

- **Class Name:**
  `UserDao`
- **Purpose:**
  To perform CRUD operations on user entities, authenticate users, and manage user security and preferences.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
criteriaList	private	List<String>	Stores criteria for filtering users.
parameterMap	private	Map<String, Object>	Stores query parameters for operations.
sb	private	StringBuilder	Builds dynamic query strings.

Methods

Method Signature	Access Modifier	Description
getQueryParam(UserCriteria criteria, FilterCriteria filterCriteria)	protected	Constructs query parameters for

		filtering users.
authenticate(String username, String password)	public	Authenticates the user based on username and password.
create(User user)	public	Creates a new user and stores it in the database.
update(User user)	public	Updates user properties in the database.
updatePassword(User user)	public	Updates the user's password securely.
getById(String id)	public	Retrieves a user by their unique ID.
getActiveByUsername(String username)	public	Retrieves an active user based on their username.
getActiveByPasswordResetKey(String passwordResetKey)	public	Retrieves a user using their password reset key.
delete(String username)	public	Soft deletes a user and their linked data.
hashPassword(String password)	protected	Hashes a password using the BCrypt algorithm.

3. Relationships

Inheritance

- **Parent Class:**

  `BaseDao<UserDto, UserCriteria>`

- **Child Classes:**

  None explicitly defined.

Associations

- **Dependency:**

    - `EntityManager`: Handles database interactions.
- **Aggregation:**
    - `ThreadLocalContext`: Provides thread-local storage for the current entity manager.
- **Composition:**
    - `QueryParam`: Encapsulates query parameters and related data.

▼ public class UserMapper

1. Class Information

- **Class Name:** `UserMapper`
- **Purpose:** Responsible for mapping raw database result sets to `UserDto` objects, encapsulating user information such as ID, username, email, creation timestamp, and locale.

2. Attributes and Methods

Attributes:

This class does not have instance attributes besides those inherited from its superclass `ResultMapper`.

Methods:

Method Signature	Access Modifier	Description
`public UserDto map(Object[] o)`	`public`	Maps an array of raw database values to a `UserDto`.

3. Relationships

Inheritance:

- **Parent Class:** `ResultMapper<UserDto>`
- **Child Classes:** None

Associations:

- **Dependency:** Depends on `UserDto` for the target mapping type.

- **Aggregation:** Not applicable
- **Composition:** Not applicable

▼ public class UserUtil

1. Class Information
    - Class Name: UserUtil
    - Purpose:
        - Provides a utility method for retrieving a user's username.
        - It ensures consistent access to user-related data.
2. Attributes and Methods
    - Attributes: NA

Name	Access Modifier	Type	Description

    - Methods:

Method Signature	Access Modifier	Description
String getUserName(User user)	public static	Returns the username of a given User object.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - User: Calls getUsername() on the provided User object.
        - Aggregation:
        - Composition:

▼ public class UserCreatedEvent

1. Class Information

- Class Name: UserCreatedEvent
- Purpose: This class encapsulates an event that is raised when a new user is created. It holds the reference to the created user and provides getter and setter methods to access or modify it.

2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
user	private	User	Stores the user object associated with the event.

   - Methods:

Method Signature	Access Modifier	Description
User getUser()	public	Returns the created User object.
void setUser(User user)	public	Sets the User object associated with this event.
String toString()	public	Returns a string representation of the UserCreatedEvent instance for debugging/logging purposes.

3. Relationships:
   - Inheritance: NA
   - Associations
     - Composition: User - The `UserCreatedEvent` class contains an instance of the `User` class, indicating a **composition** relationship. The event cannot exist meaningfully without the associated user.

▼ public class PasswordChangedEvent

1. Class Information

- Class Name: PasswordChangedEvent
- Purpose: This class is responsible for encapsulating an event that is raised when a user updates their password. It stores the `User` object associated with the password change event. It helps in tracking password changes, potentially notifying other components that depend on user authentication.

2. Attributes and Methods
   - Attributes:

Name	Access Modifier	Type	Description
user	private	User	Represents the user whose password has been changed.

   - Methods:

Method Signature	Access Modifier	Description
User getUser()	public	Returns the User object associated with the event.
void setUser(User user)	public	Sets the User object associated with the event.

3. Relationships:
   - Inheritance : NA
   - Associations
     - Dependency: User - The PasswordChangedEvent class maintains a reference to a User object, indicating which user triggered the password change event.

▼ public class AuthenticationToken

1. Class Information

- **Class Name:** `AuthenticationToken`

- **Purpose:** Represents an authentication token used for user authentication and session tracking.

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`id`	Private	String	Unique identifier for the authentication token.
`userId`	Private	String	ID of the user associated with the token.
`longLasted`	Private	boolean	Indicates if the token is for a long-lasting session.
`creationDate`	Private	Date	The date when the token was created.
`lastConnectionDate`	Private	Date	The last time this token was used for authentication.

Methods:

Method Signature	Access Modifier	Description
`public String getId()`	Public	Retrieves the token ID.
`public void setId(String id)`	Public	Sets the token ID.
`public String getUserId()`	Public	Retrieves the user ID associated with the token.
`public void setUserId(String userId)`	Public	Sets the user ID associated with the token.
`public boolean isLongLasted()`	Public	Checks if the token is for a long-lasting session.
`public void setLongLasted(boolean longLasted)`	Public	Sets the long-lasting session flag.
`public Date getCreationDate()`	Public	Retrieves the token creation date.

`public void setCreationDate(Date creationDate)`	Public	Sets the token creation date.
`public Date getLastConnectionDate()`	Public	Retrieves the last connection date using this token.
`public void setLastConnectionDate(Date lastConnectionDate)`	Public	Sets the last connection date.
`public String toString()`	Public	Provides a string representation of the object.

3. Relationships : NA

▼ public class AutheticationTokenDao

1. Class Information

- **Class Name:** `AuthenticationTokenDao`
- **Purpose:** Provides methods to create, retrieve, update, and delete authentication tokens in the database.

2. Attributes and Methods

Attributes:

This class does not define attributes but relies on `EntityManager` to interact with the database.

Methods:

Method Signature	Access Modifier	Description
`public AuthenticationToken get(String id)`	Public	Retrieves an authentication token by ID.
`public String create(AuthenticationToken authenticationToken)`	Public	Creates a new authentication token and returns its ID.
`public void delete(String authenticationTokenId) throws Exception`	Public	Deletes an authentication token by ID. Throws an exception if not found.

`public void deleteOldSessionToken(String userId)`	Public	Deletes old short-lived tokens for a user.
`public void updateLastConnectionDate(String id)`	Public	Updates the last connection date of an authentication token.

3. Relationships

Inheritance:

- **Parent Class:** None (Standalone DAO class)
- **Child Classes:** None

Associations:

- **Dependency:** Uses `EntityManager` from `ThreadLocalContext` for database operations.
- **Aggregation:** Works with `AuthenticationToken` objects.
- **Composition:** Manages the persistence lifecycle of authentication tokens.

▼ public class IPrincipal

Class Information

Class Name: `IPrincipal`

Purpose: Defines an interface representing a security principal within the application. It extends the standard `Principal` interface and provides additional methods to access user-specific information like locale, timezone, and email.

Attributes and Methods

Attributes:

- None

Methods:

Method Signature	Access Modifier	Description
`isAnonymous()`	public	Checks if the principal represents an anonymous user.
`getId()`	public	Returns the ID of the connected user, or null if the user is anonymous.
`getLocale()`	public	Returns the locale of the principal, used for localization.
`getDateTimeZone()`	public	Returns the timezone of the principal, used for date/time conversions.
`getEmail()`	public	Returns the email address of the principal.

Relationships : NA

▼ public class RoleBaseFunctionDao

1. Class Information

- **Class Name:** `RoleBaseFunctionDao`
- **Purpose:** Provides methods to retrieve the set of base functions associated with a specific role.

2. Methods and Their Details

Method Signature	**Access Modifier**	**Description**
`Set<String> findByRoleId(String roleId)`	public	Retrieves a set of base function IDs associated with a given role. Returns an empty set if no base functions are found.

- **Method Details:**
  - **Parameters:**
    - `roleId`: A string representing the role's unique identifier.
  - **Returns:** A set of base function IDs (`Set<String>`) corresponding to the specified role.
  - **Implementation Highlights:**

- Joins `T_ROLE_BASE_FUNCTION` and `T_ROLE` tables to ensure valid role and base function entries.
- Filters out logically deleted records using `DELETEDATE_D is null`.
- Uses `Sets.newHashSet()` to create a distinct result set.

---

3. Relationships

- **Dependencies:**
  - Uses `ThreadLocalContext` for `EntityManager` retrieval.

▼ public class Role

1. Class Information
   - Class Name: Role
   - Purpose:
     - Stores information about user roles (e.g., "Admin", "Editor", "Reader").
     - Used to define access control and assign permissions to users.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique identifier for the role.
name	private	String	Name of the role (e.g., "Admin", "Editor").
createDate	private	Date	Timestamp when the role was created.
deleteDate	private	Date	Timestamp when the role was deleted (if applicable).

   - Methods:

Method Signature	Access Modifier	Description
String getId()	public	Returns the **role ID**.
void setId(String id)	public	Sets the **role ID**.
String getName()	public	Returns the **role name**.
void setName(String name)	public	Sets the **role name**.
Date getCreateDate()	public	Returns the **creation date**.
void setCreateDate(Date createDate)	public	Sets the **creation date**.
Date getDeleteDate()	public	Returns the **deletion date**.
void setDeleteDate(Date deleteDate)	public	Sets the **deletion date**.
String toString()	public	Returns a **string representation** of the role.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - User: User is assigned roles
        - Aggregation:
        - Composition:

▼ public class RoleBaseFunction

1. Class Information
    - Class Name: RoleBaseFunction
    - Purpose:

- Represents a many-to-many mapping between roles (Role) and permissions (BaseFunction).
- Used to assign permissions to roles dynamically.

2. Attributes and Methods

- Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique identifier for the role-function mapping.
roleId	private	String	ID of the role that has a base function.
baseFunctionId	private	String	ID of the base function assigned to the role.
createDate	private	Date	Timestamp when the mapping was created.
deleteDate	private	Date	Timestamp when the mapping was deleted (if applicable).

- Methods:

Method Signature	Access Modifier	Description
String getId()	public	Returns the mapping ID.
void setId(String id)	public	Sets the mapping ID.
String getRoleId()	public	Returns the role ID.
void setRoleId(String roleId)	public	Sets the role ID.
String getBaseFunctionId()	public	Returns the base function ID.
void setBaseFunctionId(String baseFunctionId)	public	Sets the base function ID.

Date getCreateDate()	public	Returns the creation date.
void setCreateDate(Date createDate)	public	Sets the creation date.
Date getDeleteDate()	public	Returns the deletion date.
void setDeleteDate(Date deleteDate)	public	Sets the deletion date.
String toString()	public	Returns a string representation of the mapping.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
        - Aggregation:
            - Role: A role consists of one or more base functions that define permissions.
            - BaseFunction: A base function belongs to one or more roles, providing permissions for actions.
        - Composition:

▼ public class BaseFunction

1. Class Information
    - Class Name: BaseFunction
    - Purpose:
        - Defines system-wide roles that can be assigned to users.
        - Provides a way to manage user permissions and access control.

2. Attributes and Methods
    - Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique identifier for the function (e.g., "ADMIN").

    - Methods:

Method Signature	Access Modifier	Description
String getId()	public	Returns the function ID.
void setId(String id)	public	Sets the function ID.
String toString()	public	Returns a string representation of the function.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
        - Aggregation:
            - User: User has a BaseFunction
        - Composition:

▼ public class SecurityFilter

1. Class Information

- **Class Name:** `SecurityFilter`

- **Purpose:** Abstract filter that handles authentication and injects user identity into requests.

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`PRINCIPAL_ATTRIBUTE`	Public, Static, Final	String	Name of the request attribute containing the principal.
`LOG`	Public, Static, Final	Logger	Logger instance for the filter.

Methods:

Method Signature	Access Modifier	Description
`private static boolean hasIdentifiedUser(HttpServletRequest request)`	Private, Static	Checks if the request has an identified user.
`private static void injectUser(HttpServletRequest request, User user)`	Private, Static	Injects the user into the request with appropriate authentication state.
`private static void injectAuthenticatedUser(HttpServletRequest request, User user)`	Private, Static	Injects an authenticated user into the request attributes.
`private static void injectAnonymousUser(HttpServletRequest request)`	Private, Static	Injects an anonymous user into the request attributes.
`public void init(FilterConfig filterConfig) throws ServletException`	Public	Initializes the filter (no operation).
`public void destroy()`	Public	Destroys the filter (no operation).

`public void doFilter(ServletRequest req, ServletResponse response, FilterChain filterChain) throws IOException, ServletException`	Public	Applies the security filter by authenticating users and forwarding the request.
`protected abstract User authenticate(HttpServletRequest request)`	Protected, Abstract	Authenticates a user based on the request parameters.

3. Relationships

Inheritance:

- **Parent Class:** Implements `Filter` interface.

Associations:

- **Dependency:** Uses `User`, `UserPrincipal`, `RoleBaseFunctionDao`
- **Aggregation:** Works with `HttpServletRequest` to manage authentication.
- **Composition:** None.

▼ public class TokenBasedSecurityFilter

1. Class Information
   - Class Name: TokenBasedSecurityFilter
   - Purpose: Authenticates users using token-based authentication via cookies
2. Attributes and Methods
   - Attributes:
     - COOKIE_NAME (public static final) String: Name of authentication cookie
     - TOKEN_LONG_LIFETIME (public static final) int: Long-term token lifetime in seconds
     - TOKEN_SESSION_LIFETIME (public static final) int: Session token lifetime in seconds
   - Methods:

- extractAuthToken(Cookie[] cookies): private static String - Extracts auth token from cookies
- handleExpiredToken(AuthenticationTokenDao dao, String authTokenID): private static void - Handles expired token deletion
- isTokenExpired(AuthenticationToken authenticationToken): private static boolean - Checks token expiration
- authenticate(HttpServletRequest request): protected User - Authenticates user based on token

Inheritance

- **Parent Class:** `SecurityFilter`
- **Child Classes:** None

Associations

- **Dependency:**
  - Depends on `AuthenticationTokenDao` for retrieving, validating, and deleting authentication tokens.
  - Depends on `UserDao` for fetching user details based on authentication tokens.
  - Uses `HttpServletRequest` to extract cookies for authentication.
- **Aggregation:**
  - Aggregates `AuthenticationToken` objects retrieved from `AuthenticationTokenDao`.

▼ public class HeaderBasedSecurityFilter

Class Information

Class Name: `HeaderBasedSecurityFilter`

Purpose: A servlet filter that authenticates users based on the value of a specific HTTP header (`X-Authenticated-User`). This is typically used in environments where an external proxy handles the actual authentication

and then passes the authenticated username to the application via a header.

Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`AUTHENTICATED_USER_HEADER`	public static final	String	Constant defining the name of the HTTP header used for authentication ("X-Authenticated-User").
`enabled`	private	boolean	Flag indicating whether the filter is enabled.

Methods:

Method Signature	Access Modifier	Description
`init(FilterConfig filterConfig)`	@Override public	Initializes the filter. Reads the `enabled` flag from the filter configuration or system properties (`reader.header_authentication`).
`authenticate(HttpServletRequest request)`	@Override protected	Authenticates the user based on the `X-Authenticated-User` header. If the filter is enabled, it retrieves the username from the header and uses `UserDao` to fetch the user.

Relationships

Inheritance:

- **Parent Class:** `SecurityFilter` (Not provided, assuming it exists)

▼ public class RequestContextFilter

1. Class Information

- **Class Name:** `RequestContextFilter`
- **Purpose:** Ensures proper initialization of the request context, handles transactions, and manages logging.

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
`log`	Private, Static, Final	Logger	Logger instance for logging errors and information.

Methods:

Method Signature	Access Modifier	Description
`public void init(FilterConfig filterConfig) throws ServletException`	Public	Initializes locale, logging, and application context.
`private void addFileLogger()`	Private	Adds a file logger for application logs.
`public void destroy()`	Public	Cleans up resources (no operation).
`public void doFilter(ServletRequest request, ServletResponse response, FilterChain filterChain) throws IOException, ServletException`	Public	Manages transactions and processes the request.

3. Relationships

Inheritance:

- **Parent Class:** Implements `Filter` interface.
- **Child Classes:** None.

Associations:

- **Dependency:** Uses `EntityManager`, `EntityTransaction`, `ThreadLocalContext`, `EnvironmentUtil`, `DirectoryUtil`, and `AppContext`.
- **Composition:** None.

▼ public class UserPrincipal

1. Class Information
    - Class Name: UserPrincipal
    - Purpose: Represents an authenticated user's identity and associated information
2. Attributes and Methods
    - Attributes:
        - id (private) String: User ID
        - name (private) String: Username
        - locale (private) Locale: User's locale
        - dateTimeZone (private) DateTimeZone: User's timezone
        - email (private) String: User's email
        - baseFunctionSet (private) Set<String>: User's base functions/permissions
    - Methods:
        - Constructor(String id, String name): public
        - isAnonymous(): public boolean
        - Getters and setters for all attributes
3. Relationships
    - Inheritance:
        - Parent Interface: IPrincipal
    - Associations
        - Dependencies:
            - Locale
            - DateTimeZone

        - Set

▼ public class AnonymousPrincipal

1. Class Information

- **Class Name**: AnonymousPrincipal
- **Purpose**: Represents an anonymous user in the system with basic locale and timezone information

2. Attributes and Methods

Attributes:

- `ANONYMOUS`: public static final String - Constant defining anonymous user identifier
- `locale`: private Locale - Stores user's locale settings
- `dateTimeZone`: private DateTimeZone - Stores user's timezone settings

Methods:

- `getId()`: public String - Returns null for anonymous users
- `getName()`: public String - Returns "anonymous" constant
- `isAnonymous()`: public boolean - Always returns true
- `getLocale()`: public Locale - Returns user's locale
- `setLocale(Locale locale)`: public void - Sets user's locale
- `getDateTimeZone()`: public DateTimeZone - Returns user's timezone
- `getEmail()`: public String - Returns null for anonymous users
- `setDateTimeZone(DateTimeZone)`: public void - Sets user's timezone

3. Relationships

Inheritance:

- Parent Class: Implements IPrincipal interface
- Child Classes: None identified

Associations: None

- Aggregation: None

- Composition: None

▼ public class BaseResource

1. Class Information

- Class Name: BaseResource
- Purpose: Provides common functionality and security checks for all REST resources

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
request	protected	HttpServletRequest	Injected HTTP request
appKey	protected	String	Application key from query parameter
principal	protected	IPrincipal	Principal of authenticated user

Methods

Method Signature	Access Modifier	Description
authenticate()	protected	Checks if user is authenticated
checkBaseFunction(BaseFunction)	protected	Verifies user has specific base function
hasBaseFunction(BaseFunction)	protected	Checks if user has specific base function

3. Relationships

Inheritance:

- **Parent Class:** None (Base class)
- **Child Classes:**

Associations:

- **Dependency:**
  - Uses `SecurityFilter` to retrieve the authenticated principal.
  - Uses `BaseFunction` for role-based access control.
  - Uses `IPrincipal` and `UserPrincipal` for user identity management.
  - Uses `ForbiddenClientException` for authorization failures.
- **Aggregation:**
  - Aggregates `IPrincipal`, which represents an authenticated user.

▼ public class Locale

1. Class Information
   - Class Name: Locale
   - Purpose:
     - Defines a locale identifier (e.g., "en_US", "fr_FR").
     - Used for regional formatting preferences such as dates, currency, and translations.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
id	private	String	Unique identifier for the locale (e.g., "en_US", "fr_FR").

   - Methods:

Method Signature	Access Modifier	Description
String getId()	public	Returns the locale ID.
void setId(String id)	public	Sets the locale ID.
String toString()	public	Returns a string representation of the locale.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
        - Aggregation:
        - Composition:

▼ public class LocaleDao

1. Class Information

- **Class Name:** `LocaleDao`
- **Role in Subsystem:** User Management Subsystem
- **Purpose:** Provides methods for managing and accessing locale-related data in the system, including retrieval by ID and fetching all available locales.

---

2. Methods and Their Details

Method Signature	Access Modifier	Description
`Locale getById(String id)`	public	Retrieves a locale by its unique identifier. Returns `null` if no locale is found.
`List<Locale> findAll()`	public	Fetches all available locales sorted by their ID.

---

3. Relationships

- **Associations:**
    - Directly interacts with the `Locale` entity for database operations.
    - Utilizes `EntityManager` for persistence and query execution.
- **Dependencies:**

- Relies on `ThreadLocalContext` to access the current `EntityManager`.
- Employs the JPA API for query management.

▼ public class LocaleUtil

1. Class Information

- **Class Name:** LocaleUtil
- **Purpose:** Provides helper methods for handling locale-related operations, such as converting locale codes to `Locale` objects and extracting locale IDs from HTTP headers.

2. Attributes and Methods

Attributes:

Attribute Name	Access Modifier	Type	Description
None	-	-	This class does not define instance variables; it only provides static methods.

Methods:

Method Signature	Access Modifier	Description
`public static final Locale getLocale(String localeCode)`	`public static`	Converts a locale code (e.g., `fr_FR`) into a `Locale` instance.
`public static String getLocaleIdFromAcceptLanguage(String acceptLanguageHeader)`	`public static`	Extracts a locale ID from the `Accept-Language` header of an HTTP request.

3. Relationships

Inheritance:

- **Parent Class:** None (directly extends `Object`).

- **Child Classes:** None.

Associations:

- **Dependency:**
  - `LocaleDao`: Used to retrieve locale details from the database.
- **Aggregation:**
  - Utilizes `Locale` objects but does not own them.
- **Composition:**
  - None (does not create or destroy instances of other objects).

▼ public class LocaleResource

1. Class Information

- **Class Name:** `LocaleResource`
- **Purpose:** Provides an API endpoint to retrieve the list of all available locales in JSON format.

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
None	-	-	This class does not define any attributes.

Methods

Method Signature	Access Modifier	Description
`public Response list() throws JSONException`	`public`	Retrieves all available locales and returns them as a JSON response.

3. Relationships

Inheritance

- **Parent Class:** `BaseResource`
- **Child Classes:** None

Associations

- **Dependency:** `LocaleDao`, `Locale`
- **Aggregation:** None
- **Composition:** None

▼ public class ThemeResource

---

1. Class Information

- **Class Name:** `ThemeResource`
- **Purpose:** Provides an API endpoint to retrieve the list of all available themes in JSON format.

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
None	-	-	This class does not define any attributes.

Methods

Method Signature	Access Modifier	Description
`public Response list() throws JSONException`	`public`	Retrieves all available themes and returns them as a JSON response.

3. Relationships

Inheritance

- **Parent Class:** `BaseResource`
- **Child Classes:** None

Associations

- **Dependency:**
  - `ThemeDao` (used to fetch themes from the database)
  - `EnvironmentUtil` (used to determine if the request is from a unit test)
- **Aggregation:** None
- **Composition:** None

▼ public class ThemeDao

1. Class Information

- **Class Name:** `ThemeDao`
- **Purpose:** Provides access to theme-related data by retrieving available theme stylesheets from specified directories.

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
`STYLESHEETS_THEME_DIRS`	`public static final`	`List<String>`	Defines the directories where theme stylesheets are located.
`CSS_FILTER`	`private static final`	`FilenameFilter`	Filters files to include only `.css` and `.less` files.

Methods

Method Signature	Access Modifier	Description
`public List<String> findAll(ServletContext`	`public`	Retrieves all available themes by scanning the defined stylesheet

`servletContext)`		directories.

3. Relationships

Inheritance

- **Parent Class:** None
- **Child Classes:** None

Associations

- **Dependency:**
  - `ServletContext` (used to access theme files in a web application context)
  - `FilenameFilter` (used for filtering theme files)
- **Aggregation:** None
- **Composition:** None

▼ public class JobResource

1. Class Information

- Class Name: JobResource
- Purpose: Manages background job operations

2. Attributes and Methods

Methods

Method Signature	Access Modifier	Description
delete(String id)	public	Deletes a job

3. Relationships

Inheritance

- Parent Class: BaseResource
- Child Classes: None

Associations

Dependencies:

- JobDao
- Job entity

▼ public class TransactionUtil

1. Class Information
    - Class Name: TransactionUtil
    - Purpose:
        - Provides transaction management for database operations using JPA's EntityManager and EntityTransaction. Ensures that database changes occur within a transaction, allowing for proper rollback in case of errors.
        - Encapsulates JPA transactions within a safe execution context, ensures transactions are properly committed or rolled back to maintain data integrity, provides a utility method to manually commit the current transaction.
2. Attributes and Methods
    - Attributes:

Name	Access Modifier	Type	Description
log	private static	Logger	Logger for logging errors related to transactions.

    - Methods:

Method Signature	Access Modifier	Description
void handle(Runnable runnable)	public static	Executes a runnable within a transactional context, handling errors and rollbacks.
public static void commit()	public static	Commits the current transaction and immediately starts a new one.

3. Relationships:
    - Inheritance:
        - Parent Class:
        - Child Classes:
    - Associations
        - Dependency:
            - ThreadLocalContext: Used to retrieve the current EntityManager for database operations
            - EMF: Provides new EntityManager instances.
        - Aggregation:
        - Composition:

▼ public class JobCriteria

---

1. Class Information

- **Class Name:** `JobCriteria`
- **Purpose:**
  This class provides criteria for filtering or scheduling jobs, potentially linked to user-specific tasks in the RSS Reader system.

---

2. Attributes and Methods

Attributes

Attribute Name	Access Modifier	Type	Description
userId	private	String	Identifier for the user associated with the job criteria.

Methods

Method Signature	Access Modifier	Description
String getUserId()	public	Retrieves the user ID.
JobCriteria setUserId(String userId)	public	Sets the user ID and returns the updated instance of `JobCriteria`.

3. Relationships

- **Inheritance:**
  - **Parent Class:** None
  - **Child Classes:** None
- **Associations:**
  - **Dependency:** None explicitly defined
  - **Aggregation:** None
  - **Composition:** None

▼ public class ConfigDao

1. Class Information
   - Class Name: ConfigDao
   - Purpose: The `ConfigDao` class provides data access functionality to retrieve configuration parameters from the persistent storage by their IDs.
2. Attributes and Methods
   - Attributes:

Attribute Name	Access Modifier	Type	Description
em	private	EntityManager	Manages persistence for configuration entities

   - Methods:

Method Signature	Access Modifier	Description
Config getById(ConfigType id)	public	Retrieves a configuration parameter by its unique identifier

3. Relationships:
    - Inheritance:
        - Parent Class: None
        - Child Classes: None
    - Associations:
        - Dependency: Uses `EntityManager` to access persistent storage and manage configurations
        - Aggregation: None
        - Composition: None