

# Package: openappr (via r-universe)

October 5, 2024

**Title** Retrieve App Data from 'OpenAppBuilder'

**Version** 0.2.0

**Description** Provides an interface to connect R with the <https://github.com/IDEMSIInternational/open-app-builder> 'OpenAppBuilder' platform, enabling users to retrieve and work with user and notification data for analysis and processing. It is designed for developers and analysts to seamlessly integrate data from 'OpenAppBuilder' into R workflows via a 'Postgres' database connection, allowing direct querying and import of app data into R.

**License** LGPL (>= 3)

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**Suggests** knitr, testthat, rmarkdown

**Imports** DBI, dplyr, jsonlite, magrittr, purrr, rlang, RPostgres, RSQLite, stringr, tidyr

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Lily Clements [aut, cre]  
(<https://orcid.org/0000-0001-8864-0552>)

**Maintainer** Lily Clements <lily@idems.international>

**Date/Publication** 2024-10-04 09:00:07 UTC

**Repository** <https://lilyclements.r-universe.dev>

**RemoteUrl** <https://github.com/cran/openappr>

**RemoteRef** HEAD

**RemoteSha** e77cf29f73fb3aebc2b343931f2ce6a94c5fa4cf

## Contents

get_app_connection . . . . .	2
------------------------------	---

get_nf_data . . . . .	2
get_openapp_data . . . . .	4
get_user_data . . . . .	5
set_app_connection . . . . .	7

<b>Index</b>	<b>8</b>
--------------	----------

---

get_app_connection	<i>Get the app connection from the environment</i>
--------------------	--

---

### Description

Call the app connection. The connection is set in the function set\_app\_connection.

### Usage

```
get_app_connection()
```

### Value

returns a the app connection to the app data.

### Examples

```
# Establish a connection to the PostgreSQL database
set_app_connection(
  dbname = "vmc",
  host = "apps-server.idems.international",
  port = 5432,
  user = "vmc",
  password = "LSQkyYg5KzL747"
)

get_app_connection()
```

---

get_nf_data	<i>Get notification data from OpenAppBuilder</i>
-------------	--

---

### Description

This function retrieves data from the app\_notification\_interaction table in OpenAppBuilder and efficiently parses the notification\_meta column from JSON format.

**Usage**

```
get_nf_data(  
  site = get_app_connection(),  
  filter = FALSE,  
  filter_variable = NULL,  
  filter_variable_value = NULL  
)
```

**Arguments**

site	The name of the PostgreSQL database connection (using DBI::dbConnect or set_app_connection()).
filter	A logical value indicating whether to filter the data (defaults to FALSE).
filter_variable	A character string representing the name of the column to filter if filter == TRUE and filter_variable_value is provided.
filter_variable_value	A character string representing the value of the filter_variable column to filter if filter == TRUE.

**Value**

A data frame containing notification interaction data from OpenAppBuilder, with the notification\_meta column parsed into separate columns.

**Examples**

```
# First we need to set an app connection  
set_app_connection(  
  dbname = "vmc",  
  host = "apps-server.idems.international",  
  port = 5432,  
  user = "vmc",  
  password = "LSQkyYg5KzL747"  
)  
  
# Retrieve all notification data  
data_all_nf <- get_nf_data()  
  
# Retrieve data where 'app_user_id' is '3e68fcda-d4cd-400e-8b12-6ddfabc348'  
# or '223925c7-443a-411c-aa2a-a394f991dd52'  
valid_ids <- c("3e68fcda-d4cd-400e-8b12-6ddfabc348",  
              "223925c7-443a-411c-aa2a-a394f991dd52")  
data_filtered_users <- get_nf_data(  
  filter = TRUE,  
  filter_variable = "app_user_id",  
  filter_variable_value = valid_ids  
)
```

---

get\_openapp\_data      *Get data from OpenAppBuilder*

---

### Description

Retrieves data from OpenAppBuilder by querying the specified PostgreSQL database. The function can either retrieve all data from a specific table (e.g., app\_users or app\_notification\_interaction) or execute a custom SQL query provided by the user.

### Usage

```
get_openapp_data(
  site = get_app_connection(),
  name = c("app_users", "app_notification_interaction"),
  filter = FALSE,
  filter_variable = NULL,
  filter_variable_value = NULL,
  qry = NULL
)
```

### Arguments

site	The name of the PostgreSQL database connection (using DBI::dbConnect or set_app_connection()).
name	A character string specifying the table to retrieve data from. Default is "app_users", but "app_notification_interaction" can also be specified. This parameter is ignored if qry is provided.
filter	A logical value indicating whether to filter the data based on a specific column (defaults to FALSE).
filter_variable	A character string representing the name of the column to filter if filter == TRUE.
filter_variable_value	A character string or vector representing the value(s) of the filter_variable column to filter if filter == TRUE.
qry	An optional character string containing an SQL query. If provided, this query overrides the name parameter and allows for custom SQL to be executed.

### Value

A data frame containing the retrieved data from the specified PostgreSQL table or the result of the executed SQL query.

**Examples**

```

# First we need to set an app connection
set_app_connection(
  dbname = "vmc",
  host = "apps-server.idems.international",
  port = 5432,
  user = "vmc",
  password = "LSQkyYg5KzL747"
)

# Retrieve all data from the 'app_users' table
data_all_users <- get_openapp_data(name = "app_users")

# Retrieve filtered data from the 'app_users' table where 'app_user_id' is
# a specified ID.
valid_ids <- c("3e68fcda-d4cd-400e-8b12-6ddfabc348",
              "223925c7-443a-411c-aa2a-a394f991dd52")
data_filtered_users <- get_openapp_data(
  name = "app_users",
  filter = TRUE,
  filter_variable = "app_user_id",
  filter_variable_value = valid_ids
)

# Retrieve data using a custom SQL query
custom_query_data <- get_openapp_data(
  qry = "SELECT * FROM app_users WHERE app_version = '0.16.33'"
)

```

---

get\_user\_data

*Get user data from OpenAppBuilder*


---

**Description**

Retrieves data from the `app_users` table in `OpenAppBuilder`, and efficiently converts the `contact_fields` column from JSON format to a data frame. If `filter` is `TRUE`, the function further filters the data to include only rows where the specified `filter_variable` column matches `filter_variable_value`.

**Usage**

```

get_user_data(
  site = get_app_connection(),
  filter = FALSE,
  filter_variable = NULL,
  filter_variable_value = NULL,
  date_from = NULL,
  date_to = NULL,
  format_date = "%Y-%m-%d",
  tzone_date = "UTC"
)

```

**Arguments**

site	The name of the PostgreSQL database connection (using <code>DBI::dbConnect</code> or <code>set_app_connection()</code> ).
filter	A logical value indicating whether to filter data (defaults to <code>FALSE</code> ).
filter_variable	A character string representing the name of the column to filter if <code>filter == TRUE</code> and <code>filter_variable_value</code> is provided.
filter_variable_value	A character string representing the value of the <code>filter_variable</code> column to filter if <code>filter == TRUE</code> .
date_from	An optional character string representing the date from which to retrieve data.
date_to	An optional character string representing the date to which to retrieve data.
format_date	A character string specifying the format of the date strings (defaults to <code>"%Y-%m-%d"</code> ).
tzone_date	A character string specifying the time zone for the date strings (defaults to <code>"UTC"</code> ). System-specific (see <a href="#">as.POSIXlt</a> ), but <code>"</code> uses the current time zone, and <code>"GMT"</code> is UTC (Universal Time, Coordinated). Invalid values are most commonly treated as UTC, on some platforms with a warning.

**Value**

A data frame containing user data from the PostgreSQL database, with the `contact_fields` column parsed into separate columns.

**Examples**

```
# First we need to set an app connection
set_app_connection(
  dbname = "vmc",
  host = "apps-server.idems.international",
  port = 5432,
  user = "vmc",
  password = "LSQkyYg5KzL747"
)

# Retrieve all data from the 'app_users' table
data_all_users <- get_user_data()

# Retrieve data from the 'app_users' table where 'app_user_id' is
# a specified ID.
valid_ids <- c("3e68fcda-d4cd-400e-8b12-6ddfabc348",
              "223925c7-443a-411c-aa2a-a394f991dd52")
data_filtered_users <- get_user_data(
  filter = TRUE,
  filter_variable = "app_user_id",
  filter_variable_value = valid_ids
)
```

```
# Retrieve user data within a specific date range
date_filtered_data <- get_user_data(
  date_from = "2023-01-01",
  date_to = "2024-08-18"
)
```

---

set\_app\_connection      *Set Application Database Connection*

---

## Description

Establishes a connection to a PostgreSQL database using provided credentials. This function utilises the DBI and RPostgres packages to set up the connection.

## Usage

```
set_app_connection(dbname, host, port, user, password, ...)
```

## Arguments

dbname	The name of the database to connect to.
host	The host name of the server where the database is located.
port	The port number to connect through.
user	The username for database authentication.
password	The password for database authentication.
...	Additional arguments passed to DBI::dbConnect.

## Value

A database connection object of class DBIConnection.

## See Also

[dbConnect](#) for more details on the underlying connection function. For additional information on database interfaces, see <https://dbi.r-dbi.org/>.

## Examples

```
# Establish a connection to the PostgreSQL database
set_app_connection(
  dbname = "vmc",
  host = "apps-server.idems.international",
  port = 5432,
  user = "vmc",
  password = "LSQkyYg5KzL747"
)
```

# Index

`as.POSIXlt`, [6](#)

`dbConnect`, [7](#)

`get_app_connection`, [2](#)

`get_nf_data`, [2](#)

`get_openapp_data`, [4](#)

`get_user_data`, [5](#)

`set_app_connection`, [7](#)