

ETM

1.0

Generated by Doxygen 1.8.17

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 Class Documentation	5
3.1 Auction Class Reference	5
3.1.1 Member Function Documentation	5
3.1.1.1 endAuction()	5
3.1.1.2 getOngoingCOAuctions()	6
3.1.1.3 getOngoingDriverAuctions()	6
3.1.1.4 getOrderIDs()	6
3.1.1.5 getRunningAuctions()	7
3.1.1.6 getRunningDriverAuctions()	7
3.1.1.7 getWonCOAuctions()	8
3.1.1.8 getWonDriverAuctions()	8
3.1.1.9 hasBidder()	8
3.1.1.10 makeCOAuction()	9
3.1.1.11 makeDriverAuction()	9
3.1.1.12 setBidAmount()	10
3.1.1.13 setBidderName()	10
3.2 AuctionStatuses Struct Reference	10
3.3 CargoOwnerSignupInfo Struct Reference	11
3.4 COAuctionInfo Struct Reference	11
3.5 CourierSignupInfo Struct Reference	11
3.6 DBHandler Class Reference	12
3.6.1 Member Function Documentation	12
3.6.1.1 getResult()	12
3.6.1.2 getResult2DVector()	12
3.6.1.3 getResultVector()	13
3.6.1.4 writeFields()	13
3.7 DriverAuctionInfo Struct Reference	13
3.8 DriverSignupInfo Struct Reference	14
3.9 ForwarderSignupInfo Struct Reference	14
3.10 HomePage Class Reference	14
3.11 InfoMinLengths Struct Reference	15
3.12 Login Class Reference	15
3.12.1 Member Function Documentation	15
3.12.1.1 getUserType()	16
3.12.1.2 isValidCargoOwnerSignup()	17
3.12.1.3 isValidCourierSignup()	17
3.12.1.4 isValidDriverSignup()	17

3.12.1.5 isValidForwarderSignup()	19
3.12.1.6 isValidLogin()	19
3.12.1.7 isValidSignup()	19
3.12.1.8 storeCargoOwnerSignupDetails()	21
3.12.1.9 storeCourierSignupDetails()	21
3.12.1.10 storeDriverSignupDetails()	21
3.12.1.11 storeForwarderSignupDetails()	22
3.12.1.12 storeSignupDetails()	22
3.13 LoginInfo Struct Reference	22
3.14 LoginPage Struct Reference	23
3.15 MainWindow Class Reference	23
3.15.1 Constructor & Destructor Documentation	23
3.15.1.1 MainWindow()	23
3.15.1.2 ~MainWindow()	24
3.16 Order Class Reference	24
3.16.1 Member Function Documentation	24
3.16.1.1 deliverOrder()	24
3.16.1.2 getAllCurrentOrders()	25
3.16.1.3 getAllTakenOrders()	25
3.16.1.4 getCurrentOrders()	25
3.16.1.5 getPastOrders()	26
3.16.1.6 getTakenOrders()	26
3.16.1.7 increaseTotalPrice()	26
3.16.1.8 makeOrder()	27
3.16.1.9 takeOrder()	27
3.17 OrderInfo Struct Reference	27
3.18 OrderStatuses Struct Reference	28
3.19 RegistrationCheck Class Reference	28
3.19.1 Constructor & Destructor Documentation	28
3.19.1.1 RegistrationCheck()	28
3.19.2 Member Function Documentation	29
3.19.2.1 getResponse()	29
3.20 Secrets Class Reference	29
3.20.1 Member Function Documentation	29
3.20.1.1 getAPIKey()	29
3.20.1.2 getDBCredentials()	30
3.21 SignupInfo Struct Reference	30
3.22 UserTypes Struct Reference	30
3.23 UserUtils Class Reference	30

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Auction	5
AuctionStatuses	10
CargoOwnerSignupInfo	11
COAuctionInfo	11
CourierSignupInfo	11
DBHandler	12
DriverAuctionInfo	13
DriverSignupInfo	14
ForwarderSignupInfo	14
InfoMinLengths	15
Login	15
LoginInfo	22
LoginPage	23
Order	24
OrderInfo	27
OrderStatuses	28
QMainWindow	
MainWindow	23
QWidget	
HomePage	14
RegistrationCheck	28
Secrets	29
SignupInfo	30
UserTypes	30
UserUtils	30

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Auction	5
AuctionStatuses	10
CargoOwnerSignupInfo	11
COAuctionInfo	11
CourierSignupInfo	11
DBHandler	12
DriverAuctionInfo	13
DriverSignupInfo	14
ForwarderSignupInfo	14
HomePage	14
InfoMinLengths	15
Login	15
LoginInfo	22
LoginPage	23
MainWindow	23
Order	24
OrderInfo	27
OrderStatuses	28
RegistrationCheck	28
Secrets	29
SignupInfo	30
UserTypes	30
UserUtils	30

Chapter 3

Class Documentation

3.1 Auction Class Reference

Static Public Member Functions

- static std::vector< **COAuctionInfo** > **getRunningAuctions** (const EUserTypes &userType, const std::string &username)
- static std::vector< **COAuctionInfo** > **getOngoingCOAuctions** (const EUserTypes &userType)
- static std::vector< **DriverAuctionInfo** > **getRunningDriverAuctions** (const EUserTypes &userType, const std::string &username)
- static std::vector< **COAuctionInfo** > **getWonCOAuctions** (const std::string &username)
- static std::vector< **DriverAuctionInfo** > **getOngoingDriverAuctions** (const EUserTypes &userType)
- static std::vector< **DriverAuctionInfo** > **getWonDriverAuctions** (const std::string &username)
- static std::vector< std::string > **getOrderIDs** (const EUserTypes &userType)
- static bool **hasBidder** (const EUserTypes &userType, const std::string &auctionID)
- static void **setBidAmount** (const EUserTypes &userType, const std::string &auctionID, const double &new↵ Bid)
- static void **setBidderName** (const EUserTypes &userType, const std::string &auctionID, const std::string &username)
- static void **makeCOAuction** (const std::string &username, const std::string &orderId, const double &start↵ Price, const double &commission, const int &length)
- static void **makeDriverAuction** (const std::string &username, const std::string &orderId, const double &startPrice, const double &commission, const double &cpm, const double &distance, const int &length)
- static void **endAuction** (const EUserTypes &userType, const std::string &auctionID)

3.1.1 Member Function Documentation

3.1.1.1 endAuction()

```
void Auction::endAuction (
    const EUserTypes & userType,
    const std::string & auctionID ) [static]
```

Set the auctionStatus of the selected auction table to 'Finished' based on the specified auctionID.

Parameters

<i>userType</i>	The type of auction table.
<i>auctionID</i>	The auctionID of the auction to be finished.

3.1.1.2 getOngoingCOAuctions()

```
std::vector< COAuctionInfo > Auction::getOngoingCOAuctions (
    const EUserTypes & userType ) [static]
```

Get a list of ongoing cargo owner auctions from the database.

Parameters

<i>userType</i>	The user type.
-----------------	----------------

Returns

The list of ongoing cargo owner auctions.

3.1.1.3 getOngoingDriverAuctions()

```
std::vector< DriverAuctionInfo > Auction::getOngoingDriverAuctions (
    const EUserTypes & userType ) [static]
```

Get a list of ongoing driver auctions from the database.

Parameters

<i>userType</i>	The user type.
-----------------	----------------

Returns

The list of ongoing driver auctions.

3.1.1.4 getOrderIDs()

```
std::vector< std::string > Auction::getOrderIDs (
    const EUserTypes & userType ) [static]
```

Get a list of all the orderIDs from the specified auction table.

Parameters

<i>userType</i>	The table type.
-----------------	-----------------

Returns

The list of orderIDs.

3.1.1.5 getRunningAuctions()

```
std::vector< COAuctionInfo > Auction::getRunningAuctions (
    const EUserTypes & userType,
    const std::string & username ) [static]
```

Get a list of running auctions for the user type and username specified from the database.

Parameters

<i>userType</i>	The user type.
<i>username</i>	The username.

Returns

The list of running cargo owner auctions.

3.1.1.6 getRunningDriverAuctions()

```
std::vector< DriverAuctionInfo > Auction::getRunningDriverAuctions (
    const EUserTypes & userType,
    const std::string & username ) [static]
```

Get a list of running driver auctions for the user type and username specified from the database.

Parameters

<i>userType</i>	The user type.
<i>username</i>	The username.

Returns

The list of running driver auctions.

3.1.1.7 getWonCOAuctions()

```
std::vector< COAuctionInfo > Auction::getWonCOAuctions (
    const std::string & username ) [static]
```

Get a list of won cargo owner auctions from the database.

Parameters

<i>username</i>	The username of the bidder.
-----------------	-----------------------------

Returns

The list of won cargo owner auctions.

3.1.1.8 getWonDriverAuctions()

```
std::vector< DriverAuctionInfo > Auction::getWonDriverAuctions (
    const std::string & username ) [static]
```

Get a list of won driver auctions from the database.

Parameters

<i>username</i>	The username of the bidder.
-----------------	-----------------------------

Returns

The list of won driver auctions.

3.1.1.9 hasBidder()

```
bool Auction::hasBidder (
    const EUserTypes & userType,
    const std::string & auctionID ) [static]
```

Check if the specified auctionID has a bidder.

Parameters

<i>userType</i>	The type of auction table.
<i>auctionID</i>	The auctionID of the auction to be checked.

Returns

True if the auction has a bidder, false otherwise.

3.1.1.10 makeCOAuction()

```
void Auction::makeCOAuction (
    const std::string & username,
    const std::string & orderId,
    const double & startPrice,
    const double & commission,
    const int & length ) [static]
```

Create a new auction on the COAuction table in the database.

Parameters

<i>username</i>	The name of the auction owner.
<i>orderId</i>	The orderID of the cargo.
<i>startPrice</i>	The start price of the auction.
<i>commission</i>	The commission of the auction.
<i>length</i>	The length of the auction in hours.

3.1.1.11 makeDriverAuction()

```
void Auction::makeDriverAuction (
    const std::string & username,
    const std::string & orderId,
    const double & startPrice,
    const double & commission,
    const double & cpm,
    const double & distance,
    const int & length ) [static]
```

Create a new auction on the DriverAuction table in the database.

Parameters

<i>username</i>	The name of the auction owner.
<i>orderId</i>	The orderID of the cargo.
<i>startPrice</i>	The start price of the auction.
<i>commission</i>	The commission of the auction.
<i>cpm</i>	The cost per mile of the auction.
<i>distance</i>	The distance of the auction in miles.
<i>length</i>	The length of the auction in hours.

3.1.1.12 setBidAmount()

```
void Auction::setBidAmount (
    const EUserTypes & userType,
    const std::string & auctionID,
    const double & newBid ) [static]
```

Update the bid amount of the selected auction table based on the specified auctionID.

Parameters

<i>userType</i>	The type of auction table.
<i>auctionID</i>	The auctionID of the auction to be updated.
<i>newBid</i>	The new bid amount.

3.1.1.13 setBidderName()

```
void Auction::setBidderName (
    const EUserTypes & userType,
    const std::string & auctionID,
    const std::string & username ) [static]
```

Set the bidderName of the selected auction table based on the specified auctionID.

Parameters

<i>userType</i>	The type of auction table.
<i>auctionID</i>	The auctionID of the auction to be updated.
<i>username</i>	The username of the bidder.

The documentation for this class was generated from the following files:

- ETM/Application/Auction.h
- ETM/Application/Auction.cpp

3.2 AuctionStatuses Struct Reference

Public Attributes

- std::string **Running** = "Running"
- std::string **Finished** = "Finished"

The documentation for this struct was generated from the following file:

- ETM/Application/Auction.h

3.3 CargoOwnerSignupInfo Struct Reference

Public Attributes

- std::string **goodsCategory**

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.4 COAuctionInfo Struct Reference

Public Attributes

- std::string **auctionId**
- std::string **orderId**
- std::string **username**
- double **startPrice**
- double **bidPrice**
- double **commission**
- std::string **startDate**
- std::string **startTime**
- int **length**
- std::string **status**
- std::string **bidder**

The documentation for this struct was generated from the following file:

- ETM/Application/Auction.h

3.5 CourierSignupInfo Struct Reference

Public Attributes

- std::string **companyName**
- std::string **companyPhone**
- std::string **companyAddress**
- std::string **companyCity**

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.6 DBHandler Class Reference

Static Public Member Functions

- static std::string **getResult** (const std::string &query)
- static std::vector< std::string > **getResultVector** (const std::string &query)
- static std::vector< std::vector< std::string > > **getResult2DVector** (const std::string &query)
- static void **writeFields** (const std::string &query)

3.6.1 Member Function Documentation

3.6.1.1 getResult()

```
std::string DBHandler::getResult (
    const std::string & query ) [static]
```

Query the database for the given query and expect a single string result.

Parameters

<i>query</i>	The query to execute.
--------------	-----------------------

Returns

The result of the query.

3.6.1.2 getResult2DVector()

```
std::vector< std::vector< std::string > > DBHandler::getResult2DVector (
    const std::string & query ) [static]
```

Query the database for the given query and expect a whole table.

Parameters

<i>query</i>	The query to execute.
--------------	-----------------------

Returns

The table of results.

3.6.1.3 getResultVector()

```
std::vector< std::string > DBHandler::getResultVector (
    const std::string & query ) [static]
```

Query the database for the given query and expect a row result.

Parameters

<i>query</i>	The query to execute.
--------------	-----------------------

Returns

The row of results.

3.6.1.4 writeFields()

```
void DBHandler::writeFields (
    const std::string & query ) [static]
```

Write data to the database given the query.

Parameters

<i>query</i>	The query to execute.
--------------	-----------------------

The documentation for this class was generated from the following files:

- ETM/Database/DBHandler.h
- ETM/Database/DBHandler.cpp

3.7 DriverAuctionInfo Struct Reference

Public Attributes

- std::string **auctionId**
- std::string **orderId**
- std::string **username**
- double **startPrice**
- double **bidPrice**
- double **commission**
- double **costPerMile**
- double **distance**
- std::string **startDate**
- std::string **startTime**
- int **length**

- std::string **status**
- std::string **bidder**

The documentation for this struct was generated from the following file:

- ETM/Application/Auction.h

3.8 DriverSignupInfo Struct Reference

Public Attributes

- std::string **NINumber**
- std::string **drivingLicenceID**
- std::string **lorryType**
- std::string **lorryReg**
- std::string **companyAddress**
- std::string **companyCity**

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.9 ForwarderSignupInfo Struct Reference

Public Attributes

- std::string **companyName**
- std::string **companyPhone**
- std::string **companyAddress**
- std::string **companyCity**

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.10 HomePage Class Reference

Inheritance diagram for HomePage:

3.11 InfoMinLengths Struct Reference

Public Attributes

- const int **USERNAME** = 2
- const int **PASSWORD** = 0
- const int **EMAIL** = 2
- const int **FIRSTNAME** = 2
- const int **LASTNAME** = 2
- const int **AGE** = 17
- const int **PHONE** = 9
- const int **HOME_ADDRESS** = 2
- const int **HOME_CITY** = 2
- const int **COMPANY_NAME** = 2
- const int **COMPANY_PHONE** = 9
- const int **COMPANY_ADDRESS** = 2
- const int **COMPANY_CITY** = 2
- const int **NI_NUMBER** = 8
- const int **DRIVING_LICENCE_ID** = 15
- const int **LORRY_TYPE** = 1
- const int **LORRY_REG** = 3
- const int **GOODS_CATEGORY** = 1

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.12 Login Class Reference

Static Public Member Functions

- static ErrorTypes **isValidLogin** (const **LoginInfo** &loginInfo)
- static ErrorTypes **isValidSignup** (const **SignupInfo** &signupInfo)
- static bool **isValidDriverSignup** (const **DriverSignupInfo** &signupInfo)
- static bool **isValidCourierSignup** (const **CourierSignupInfo** &signupInfo)
- static bool **isValidForwarderSignup** (const **ForwarderSignupInfo** &signupInfo)
- static bool **isValidCargoOwnerSignup** (const **CargoOwnerSignupInfo** &signupInfo)
- static void **storeSignupDetails** (const **SignupInfo** &signupInfo)
- static void **storeDriverSignupDetails** (const **SignupInfo** &username, const **DriverSignupInfo** &signupInfo)
- static void **storeCourierSignupDetails** (const **SignupInfo** &username, const **CourierSignupInfo** &signupInfo)
- static void **storeForwarderSignupDetails** (const **SignupInfo** &username, const **ForwarderSignupInfo** &signupInfo)
- static void **storeCargoOwnerSignupDetails** (const **SignupInfo** &username, const **CargoOwnerSignupInfo** &signupInfo)
- static EUserTypes **getUserType** (const **LoginInfo** &loginInfo)

3.12.1 Member Function Documentation

3.12.1.1 getUserType()

```
EUserTypes Login::getUserType (  
    const LoginInfo & loginInfo ) [static]
```

Get the type of the user from the provided login info from the database.

Parameters

<i>loginInfo</i>	The login information to get the type of the user from.
------------------	---

Returns

The type of the user.

3.12.1.2 isValidCargoOwnerSignup()

```
bool Login::isValidCargoOwnerSignup (
    const CargoOwnerSignupInfo & signupInfo ) [static]
```

Check if the provided cargo owner-specific signup info is valid.

Parameters

<i>signupInfo</i>	The signup information to check.
-------------------	----------------------------------

Returns

True if the information is valid, false otherwise.

3.12.1.3 isValidCourierSignup()

```
bool Login::isValidCourierSignup (
    const CourierSignupInfo & signupInfo ) [static]
```

Check if the provided courier-specific signup info is valid.

Parameters

<i>signupInfo</i>	The signup information to check.
-------------------	----------------------------------

Returns

True if the information is valid, false otherwise.

3.12.1.4 isValidDriverSignup()

```
bool Login::isValidDriverSignup (
    const DriverSignupInfo & signupInfo ) [static]
```

Check if the provided driver-specific signup info is valid.

Parameters

<i>signupInfo</i>	The signup information to check.
-------------------	----------------------------------

Returns

True if the information is valid, false otherwise.

3.12.1.5 isValidForwarderSignup()

```
bool Login::isValidForwarderSignup (
    const ForwarderSignupInfo & signupInfo ) [static]
```

Check if the provided forwarder-specific signup info is valid.

Parameters

<i>signupInfo</i>	The signup information to check.
-------------------	----------------------------------

Returns

True if the information is valid, false otherwise.

3.12.1.6 isValidLogin()

```
ErrorTypes Login::isValidLogin (
    const LoginInfo & loginInfo ) [static]
```

Check if the provided username and password are valid.

Parameters

<i>loginInfo</i>	The login information to check.
------------------	---------------------------------

Returns

The type of error result (if none, return SUCCESS).

3.12.1.7 isValidSignup()

```
ErrorTypes Login::isValidSignup (
    const SignupInfo & signupInfo ) [static]
```

Check if the provided signup info is valid.

Parameters

<i>signupInfo</i>	The signup information to check.
-------------------	----------------------------------

Returns

The type of error result (if none, return SUCCESS).

3.12.1.8 storeCargoOwnerSignupDetails()

```
void Login::storeCargoOwnerSignupDetails (
    const SignupInfo & username,
    const CargoOwnerSignupInfo & signupInfo ) [static]
```

Store the cargo owner signup info into the database.

Parameters

<i>username</i>	The username of the cargo owner.
<i>signupInfo</i>	The signup information to store.

3.12.1.9 storeCourierSignupDetails()

```
void Login::storeCourierSignupDetails (
    const SignupInfo & username,
    const CourierSignupInfo & signupInfo ) [static]
```

Store the courier signup info into the database.

Parameters

<i>username</i>	The username of the courier.
<i>signupInfo</i>	The signup information to store.

3.12.1.10 storeDriverSignupDetails()

```
void Login::storeDriverSignupDetails (
    const SignupInfo & username,
    const DriverSignupInfo & signupInfo ) [static]
```

Store the driver signup info into the database.

Parameters

<i>username</i>	The username of the driver.
<i>signupInfo</i>	The signup information to store.

3.12.1.11 storeForwarderSignupDetails()

```
void Login::storeForwarderSignupDetails (
    const SignupInfo & username,
    const ForwarderSignupInfo & signupInfo ) [static]
```

Store the forwarder signup info into the database.

Parameters

<i>username</i>	The username of the forwarder.
<i>signupInfo</i>	The signup information to store.

3.12.1.12 storeSignupDetails()

```
void Login::storeSignupDetails (
    const SignupInfo & signupInfo ) [static]
```

Store the signup info into the database.

Parameters

<i>signupInfo</i>	The signup information to store.
-------------------	----------------------------------

The documentation for this class was generated from the following files:

- ETM/Application/Login.h
- ETM/Application/Login.cpp

3.13 LoginInfo Struct Reference**Public Attributes**

- std::string **username** = ""
- std::string **password**

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.14 LoginPage Struct Reference

Public Attributes

- const int **LOGIN** = 0
- const int **SIGNUP** = 1
- const int **SIGNUP_DRIVER** = 2
- const int **SIGNUP_CARGO_OWNER** = 3
- const int **SIGNUP_FORWARDER** = 4
- const int **SIGNUP_COURIER** = 5

The documentation for this struct was generated from the following file:

- ETM/mainwindow.h

3.15 MainWindow Class Reference

Inheritance diagram for MainWindow:

Collaboration diagram for MainWindow:

Public Member Functions

- **MainWindow** (QWidget *parent=nullptr)
- **~MainWindow** ()
- std::string **getUsername** ()
- **LoginInfo** **getLoginInfo** ()
- **SignupInfo** **getSignupInfo** ()
- **DriverSignupInfo** **getDriverSignupInfo** ()
- **CargoOwnerSignupInfo** **getCargoOwnerSignupInfo** ()
- **CourierSignupInfo** **getCourierSignupInfo** ()
- **ForwarderSignupInfo** **getForwarderSignupInfo** ()

3.15.1 Constructor & Destructor Documentation

3.15.1.1 MainWindow()

```
MainWindow::MainWindow (  
    QWidget * parent = nullptr )
```

Constructor for the main window.

Parameters

<i>parent</i>	
---------------	--

3.15.1.2 ~MainWindow()

```
MainWindow::~MainWindow ( )
```

Destructor for the main window.

The documentation for this class was generated from the following files:

- ETM/mainwindow.h
- ETM/mainwindow.cpp

3.16 Order Class Reference**Static Public Member Functions**

- static std::vector< **OrderInfo** > **getPastOrders** (const std::string &username)
- static std::vector< **OrderInfo** > **getCurrentOrders** (const std::string &username)
- static std::vector< **OrderInfo** > **getAllCurrentOrders** ()
- static std::vector< **OrderInfo** > **getTakenOrders** (const EUserTypes &userType, const std::string &username)
- static std::vector< **OrderInfo** > **getAllTakenOrders** (const EUserTypes &userType)
- static void **increaseTotalPrice** (const std::string &orderId, const double &income)
- static void **makeOrder** (const std::string &username, const std::string &itemName, const int &quantity, const double &unitPrice)
- static void **takeOrder** (const EUserTypes &userType, const std::string &username, const std::string &orderId)
- static void **deliverOrder** (const std::string &orderId)

3.16.1 Member Function Documentation**3.16.1.1 deliverOrder()**

```
void Order::deliverOrder (
    const std::string & orderId ) [static]
```

Deliver an order by the given order ID.

Parameters

<i>orderId</i>	The order ID.
----------------	---------------

3.16.1.2 getAllCurrentOrders()

```
std::vector< OrderInfo > Order::getAllCurrentOrders ( ) [static]
```

Get all the current orders from the Orders table.

Returns

A vector of order info.

3.16.1.3 getAllTakenOrders()

```
std::vector< OrderInfo > Order::getAllTakenOrders (
    const EUserTypes & userType ) [static]
```

Get all the taken orders from the Orders table.

Parameters

<i>userType</i>	The user type.
-----------------	----------------

Returns

3.16.1.4 getCurrentOrders()

```
std::vector< OrderInfo > Order::getCurrentOrders (
    const std::string & username ) [static]
```

Get current orders given the username.

Parameters

<i>username</i>	The username.
-----------------	---------------

Returns

A vector of order info.

3.16.1.5 getPastOrders()

```
std::vector< OrderInfo > Order::getPastOrders (
    const std::string & username ) [static]
```

Get the the past orders of the customer.

Parameters

<i>username</i>	The username of the customer.
-----------------	-------------------------------

Returns

The past orders.

3.16.1.6 getTakenOrders()

```
std::vector< OrderInfo > Order::getTakenOrders (
    const EUserTypes & userType,
    const std::string & username ) [static]
```

Get all the taken orders from the Orders table.

Parameters

<i>userType</i>	The user type.
<i>username</i>	The username.

Returns

A vector of order info.

3.16.1.7 increaseTotalPrice()

```
void Order::increaseTotalPrice (
    const std::string & orderID,
    const double & income ) [static]
```

Increase the total price of the orderID by the given amount.

Parameters

<i>orderID</i>	The order ID.
<i>income</i>	The amount to increase the total price by.

3.16.1.8 makeOrder()

```
void Order::makeOrder (
    const std::string & username,
    const std::string & itemName,
    const int & quantity,
    const double & unitPrice ) [static]
```

Make a new order in the Orders table given the info.

Parameters

<i>username</i>	The username of the customer.
<i>itemName</i>	The item name.
<i>quantity</i>	The quantity of the item.
<i>unitPrice</i>	The price of each item.

3.16.1.9 takeOrder()

```
void Order::takeOrder (
    const EUserTypes & userType,
    const std::string & username,
    const std::string & orderID ) [static]
```

Take an order by the given order ID and username of the user taking it.

Parameters

<i>userType</i>	The user type of the user taking the order.
<i>username</i>	The username of the user taking the order.
<i>orderID</i>	The order ID.

The documentation for this class was generated from the following files:

- ETM/Application/Order.h
- ETM/Application/Order.cpp

3.17 OrderInfo Struct Reference

Public Attributes

- std::string **id**
- std::string **username**

- `std::string` **date**
- `std::string` **time**
- `std::string` **status**
- `std::string` **itemName**
- `double` **unitPrice**
- `int` **quantity**
- `double` **fees**

The documentation for this struct was generated from the following file:

- `ETM/Application/Order.h`

3.18 OrderStatuses Struct Reference

Public Attributes

- `std::string` **Pending** = "Pending"
- `std::string` **Delivered** = "Delivered"
- `std::string` **Cancelled** = "Cancelled"

The documentation for this struct was generated from the following file:

- `ETM/Application/Order.h`

3.19 RegistrationCheck Class Reference

Public Member Functions

- **RegistrationCheck** (`std::string` `_registrationNumber`)
- `bool` **getResponse** () `const`

3.19.1 Constructor & Destructor Documentation

3.19.1.1 RegistrationCheck()

```
RegistrationCheck::RegistrationCheck (
    std::string _registrationNumber ) [explicit]
```

Constructor for the **RegistrationCheck** (p. 28) class.

Parameters

<code>_registrationNumber</code>	
----------------------------------	--

3.19.2 Member Function Documentation

3.19.2.1 `getResponse()`

```
bool RegistrationCheck::getResponse ( ) const
```

Getter for the isValid variable.

Returns

isValid

The documentation for this class was generated from the following files:

- ETM/Application/RegistrationCheck.h
- ETM/Application/RegistrationCheck.cpp

3.20 Secrets Class Reference

Static Public Member Functions

- static std::string `getAPIKey()`
- static std::string `getDBCredentials()`

3.20.1 Member Function Documentation

3.20.1.1 `getAPIKey()`

```
static std::string Secrets::getAPIKey ( ) [inline], [static]
```

Get the registration API key.

Returns

API key.

3.20.1.2 getDBCredentials()

```
static std::string Secrets::getDBCredentials ( ) [inline], [static]
```

Get the credentials to access the database.

Returns

Database credentials.

The documentation for this class was generated from the following file:

- ETM/Application/Secrets.h

3.21 SignupInfo Struct Reference

Public Attributes

- std::string **username**
- std::string **password**
- std::string **email**
- std::string **firstName**
- std::string **lastName**
- int **age**
- std::string **phone**
- std::string **homeAddress**
- std::string **homeCity**
- std::string **type**

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.22 UserTypes Struct Reference

Public Attributes

- const std::string **CargoOwner** = "CargoOwner"
- const std::string **Forwarder** = "Forwarder"
- const std::string **Driver** = "Driver"
- const std::string **Consignee** = "Consignee"
- const std::string **Admin** = "Admin"
- const std::string **Courier** = "Courier"

The documentation for this struct was generated from the following file:

- ETM/Users/UserUtils.h

3.23 UserUtils Class Reference

The documentation for this class was generated from the following file:

- ETM/Users/UserUtils.h

Index

~MainWindow
 MainWindow, 24

Auction, 5
 endAuction, 5
 getOngoingCOAuctions, 6
 getOngoingDriverAuctions, 6
 getOrderIDs, 6
 getRunningAuctions, 7
 getRunningDriverAuctions, 7
 getWonCOAuctions, 7
 getWonDriverAuctions, 8
 hasBidder, 8
 makeCOAuction, 9
 makeDriverAuction, 9
 setBidAmount, 9
 setBidderName, 10
AuctionStatuses, 10

CargoOwnerSignupInfo, 11
COAuctionInfo, 11
CourierSignupInfo, 11

DBHandler, 12
 getResult, 12
 getResult2DVector, 12
 getResultVector, 12
 writeFields, 13

deliverOrder
 Order, 24

DriverAuctionInfo, 13
DriverSignupInfo, 14

endAuction
 Auction, 5

ForwarderSignupInfo, 14

getAllCurrentOrders
 Order, 25

getAllTakenOrders
 Order, 25

getAPIKey
 Secrets, 29

getCurrentOrders
 Order, 25

getDBCredentials
 Secrets, 29

getOngoingCOAuctions
 Auction, 6

getOngoingDriverAuctions

 Auction, 6
getOrderIDs
 Auction, 6
getPastOrders
 Order, 25
getResponse
 RegistrationCheck, 29
getResult
 DBHandler, 12
getResult2DVector
 DBHandler, 12
getResultVector
 DBHandler, 12
getRunningAuctions
 Auction, 7
getRunningDriverAuctions
 Auction, 7
getTakenOrders
 Order, 26
getUserType
 Login, 15
getWonCOAuctions
 Auction, 7
getWonDriverAuctions
 Auction, 8

hasBidder
 Auction, 8
HomePage, 14

increaseTotalPrice
 Order, 26
InfoMinLengths, 15
isValidCargoOwnerSignup
 Login, 17
isValidCourierSignup
 Login, 17
isValidDriverSignup
 Login, 17
isValidForwarderSignup
 Login, 19
isValidLogin
 Login, 19
isValidSignup
 Login, 19

Login, 15
 getUserType, 15
 isValidCargoOwnerSignup, 17
 isValidCourierSignup, 17

- isValidDriverSignup, 17
- isValidForwarderSignup, 19
- isValidLogin, 19
- isValidSignup, 19
- storeCargoOwnerSignupDetails, 21
- storeCourierSignupDetails, 21
- storeDriverSignupDetails, 21
- storeForwarderSignupDetails, 22
- storeSignupDetails, 22
- LoginInfo, 22
- LoginPage, 23
- MainWindow, 23
 - ~MainWindow, 24
 - MainWindow, 23
- makeCOAuction
 - Auction, 9
- makeDriverAuction
 - Auction, 9
- makeOrder
 - Order, 27
- Order, 24
 - deliverOrder, 24
 - getAllCurrentOrders, 25
 - getAllTakenOrders, 25
 - getCurrentOrders, 25
 - getPastOrders, 25
 - getTakenOrders, 26
 - increaseTotalPrice, 26
 - makeOrder, 27
 - takeOrder, 27
- OrderInfo, 27
- OrderStatuses, 28
- RegistrationCheck, 28
 - getResponse, 29
 - RegistrationCheck, 28
- Secrets, 29
 - getAPIKey, 29
 - getDBCredentials, 29
- setBidAmount
 - Auction, 9
- setBidderName
 - Auction, 10
- SignupInfo, 30
- storeCargoOwnerSignupDetails
 - Login, 21
- storeCourierSignupDetails
 - Login, 21
- storeDriverSignupDetails
 - Login, 21
- storeForwarderSignupDetails
 - Login, 22
- storeSignupDetails
 - Login, 22
- takeOrder
 - Order, 27
- UserTypes, 30
- UserUtils, 30
- writeFields
 - DBHandler, 13