

EasyTicket

Generated by Doxygen 1.9.1

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 Class Documentation	5
3.1 Admin Class Reference	5
3.1.1 Constructor & Destructor Documentation	5
3.1.1.1 Admin()	5
3.1.2 Member Function Documentation	6
3.1.2.1 estUnAdmin()	6
3.2 Categorie Class Reference	6
3.2.1 Constructor & Destructor Documentation	6
3.2.1.1 Categorie()	6
3.3 Client Class Reference	7
3.3.1 Constructor & Destructor Documentation	7
3.3.1.1 Client()	7
3.3.2 Member Function Documentation	7
3.3.2.1 estUnClient()	8
3.4 Ingenieur Class Reference	8
3.4.1 Constructor & Destructor Documentation	8
3.4.1.1 Ingenieur()	8
3.4.2 Member Function Documentation	9
3.4.2.1 estUnIngenieur()	9
3.5 Logiciel Class Reference	9
3.5.1 Constructor & Destructor Documentation	9
3.5.1.1 Logiciel()	9
3.6 MainWindow Class Reference	10
3.7 Message Class Reference	10
3.7.1 Member Function Documentation	11
3.7.1.1 getDate_envoie()	11
3.7.1.2 getId_message()	11
3.7.1.3 getTicket()	11
3.7.1.4 getUser()	11
3.7.1.5 setDate_envoie()	12
3.7.1.6 setId_message()	12
3.7.1.7 setTicket()	12
3.7.1.8 setUser()	13
3.8 PageLogin Class Reference	13
3.9 Personnel Class Reference	14
3.9.1 Constructor & Destructor Documentation	14
3.9.1.1 Personnel()	14

3.9.2 Member Function Documentation	15
3.9.2.1 estUnIngenieur()	15
3.9.2.2 estUnPersonnel()	15
3.9.2.3 estUnTechnicien()	15
3.10 Systeme Class Reference	16
3.10.1 Constructor & Destructor Documentation	16
3.10.1.1 Systeme()	16
3.10.2 Member Function Documentation	16
3.10.2.1 getId_systeme()	16
3.10.2.2 getNom()	17
3.10.2.3 setId_systeme()	17
3.10.2.4 setNom()	17
3.11 Technicien Class Reference	17
3.11.1 Constructor & Destructor Documentation	18
3.11.1.1 Technicien()	18
3.11.2 Member Function Documentation	18
3.11.2.1 estUnTechnicien()	19
3.12 Ticket Class Reference	19
3.12.1 Constructor & Destructor Documentation	20
3.12.1.1 Ticket()	20
3.12.2 Member Function Documentation	20
3.12.2.1 getClient()	20
3.12.2.2 getDate_creation()	20
3.12.2.3 getDate_fermeture()	21
3.12.2.4 getId_ticket()	21
3.12.2.5 getListeMessages()	21
3.12.2.6 getLogiciel()	21
3.12.2.7 getPersonnel()	22
3.12.2.8 getSysteme()	22
3.12.2.9 setClient()	22
3.12.2.10 setDate_creation()	22
3.12.2.11 setDate_fermeture()	23
3.12.2.12 setId_ticket()	23
3.12.2.13 setListeMessages()	23
3.12.2.14 setLogiciel()	23
3.12.2.15 setPersonnel()	24
3.12.2.16 setSysteme()	24
3.13 Utilisateur Class Reference	24
3.13.1 Constructor & Destructor Documentation	25
3.13.1.1 Utilisateur()	25
3.13.2 Member Function Documentation	26
3.13.2.1 estUnAdmin()	26

3.13.2.2 estUnClient()	26
3.13.2.3 estUnPersonnel()	26
3.13.2.4 getId_users()	27
3.13.2.5 getMail()	27
3.13.2.6 getMdp()	27
3.13.2.7 getNom()	27
3.13.2.8 getPrenom()	28
3.13.2.9 setId_users()	28
3.13.2.10 setMail()	28
3.13.2.11 setMdp()	28
3.13.2.12 setNom()	29
3.13.2.13 setPrenom()	29

Index	31
--------------	-----------

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Categorie	6
Logiciel	9
Message	10
QMainWindow	
MainWindow	10
QWidget	
PageLogin	13
Systeme	16
Ticket	19
Utilisateur	24
Admin	5
Client	7
Personnel	14
Ingenieur	8
Technicien	17

Chapter 2

Class Index

2.1 Class List

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

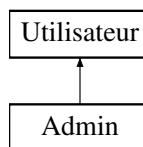
Admin	5
Categorie	6
Client	7
Ingenieur	8
Logiciel	9
MainWindow	10
Message	10
PageLogin	13
Personnel	14
Systeme	16
Technicien	17
Ticket	19
Utilisateur	24

Chapter 3

Class Documentation

3.1 Admin Class Reference

Inheritance diagram for Admin:



Public Member Functions

- [Admin](#) ()
La classe [Admin](#) permet de décrire un Administrateur dans l'application EasyTicket.
- **Admin** (std::string id_user, std::string nom, std::string prenom, std::string mdp, std::string id, std::string mail)
- bool [estUnAdmin](#) ()
Cette méthode permet de vérifier si un [Utilisateur](#) est un [Admin](#).

3.1.1 Constructor & Destructor Documentation

3.1.1.1 Admin()

```
Admin::Admin ( )
```

La classe [Admin](#) permet de décrire un Administrateur dans l'application EasyTicket.

Parameters

<i>id_user</i>	l'identifiant de l'administrateur
<i>nom</i>	le nom de l'administrateur
<i>prenom</i>	le prenom de l'administrateur
<i>mail</i>	le mail de l'administrateur

3.1.2 Member Function Documentation

3.1.2.1 estUnAdmin()

```
bool Admin::estUnAdmin ( )
```

Cette méthode permet de vérifier si un [Utilisateur](#) est un [Admin](#).

Returns

true

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

- include/admin.h
- src/admin.cpp

3.2 Categorie Class Reference

Public Member Functions

- [Categorie](#) (std::string id_categorie, std::string nom_categorie)
la classe [Categorie](#) permet de décrire une catégorie de ticket dans l'application EasyTicket.

3.2.1 Constructor & Destructor Documentation

3.2.1.1 Categorie()

```
Categorie::Categorie (
    std::string id_categorie,
    std::string nom_categorie )
```

la classe [Categorie](#) permet de décrire une catégorie de ticket dans l'application EasyTicket.

Parameters

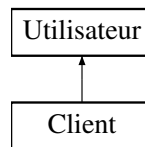
<i>id_categorie</i>	l'identifiant de la catégorie
<i>nom_categorie</i>	le nom de la catégorie

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

- include/categorie.h
- src/categorie.cpp

3.3 Client Class Reference

Inheritance diagram for Client:



Public Member Functions

- [Client](#) (std::string id_u, std::string n, std::string p, std::string motdepasse, std::string email)
La classe [Client](#) permet de décrire un [Client](#) dans l'application EasyTicket.
- bool [estUnClient](#) ()
Cette méthode permet de vérifier si un [Utilisateur](#) est un [Client](#).

3.3.1 Constructor & Destructor Documentation

3.3.1.1 Client()

```
Client::Client (
    std::string id_u,
    std::string n,
    std::string p,
    std::string motdepasse,
    std::string email )
```

La classe [Client](#) permet de décrire un [Client](#) dans l'application EasyTicket.

Parameters

<i>id_u</i>	L'identifiant de l'utilisateur
<i>n</i>	Le nom de l'utilisateur
<i>p</i>	Le prénom de l'utilisateur
<i>motdepasse</i>	Le mot de passe de l'utilisateur
<i>email</i>	Le mail de l'utilisateur

3.3.2 Member Function Documentation

3.3.2.1 estUnClient()

```
bool Client::estUnClient ( )
```

Cette méthode permet de vérifier si un [Utilisateur](#) est un [Client](#).

Returns

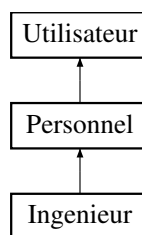
true

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

- include/client.h
- src/client.cpp

3.4 Ingenieur Class Reference

Inheritance diagram for Ingenieur:



Public Member Functions

- [Ingenieur](#) (std::string id_u, std::string n, std::string p, std::string motdepasse, std::string email)
La classe [Ingenieur](#) permet de décrire un ingénieur dans l'application EasyTicket.
- bool [estUnIngenieur](#) ()
Cette méthode permet de vérifier si un [Personnel](#) est un Ingénieur.

3.4.1 Constructor & Destructor Documentation

3.4.1.1 Ingenieur()

```
Ingenieur::Ingenieur (
    std::string id_u,
    std::string n,
    std::string p,
    std::string motdepasse,
    std::string email )
```

La classe [Ingenieur](#) permet de décrire un ingénieur dans l'application EasyTicket.

Parameters

<i>id_u</i>	l'identifiant de l'ingénieur
<i>n</i>	le nom de l'ingénieur
<i>p</i>	le prénom de l'ingénieur
<i>motdepasse</i>	le mot de passe de l'ingénieur
<i>email</i>	l'email de l'ingénieur

3.4.2 Member Function Documentation

3.4.2.1 estUnIngenieur()

```
bool Ingenieur::estUnIngenieur ( )
```

Cette méthode permet de vérifier si un [Personnel](#) est un Ingénieur.

Returns

true

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

- include/ingenieur.h
- src/ingenieur.cpp

3.5 Logiciel Class Reference

Public Member Functions

- [Logiciel](#) (std::string id_logiciel, std::string nom)
La classe logiciel permet de décrire un logiciel dans l'application EasyTicket.

3.5.1 Constructor & Destructor Documentation

3.5.1.1 Logiciel()

```
Logiciel::Logiciel (
    std::string id_logiciel,
    std::string nom )
```

La classe logiciel permet de décrire un logiciel dans l'application EasyTicket.

Parameters

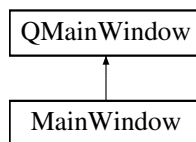
<i>id_logiciel</i>	l'identifiant du logiciel
<i>nom</i>	le nom du logiciel

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

- include/logiciel.h
- src/logiciel.cpp

3.6 MainWindow Class Reference

Inheritance diagram for MainWindow:



Public Member Functions

- **MainWindow** (QWidget *parent=nullptr)

Protected Attributes

- QStackedWidget * **stack**
- [PageLogin](#) * **page_login**

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

- include/mainwindow.h
- src/mainwindow.cpp

3.7 Message Class Reference

Public Member Functions

- [Message](#) (std::string id_message, double date_envois, [Utilisateur](#) user, [Ticket](#) ticket)
Le constructeur par défaut.
- std::string [getId_message](#) () const
Cette méthode permet de récupérer l'identifiant du message.
- void [setId_message](#) (const std::string &value)
Cette méthode permet de définir l'identifiant du message.
- double [getDate_envois](#) () const

- Cette méthode permet de récupérer la date à laquelle le message a été envoyé.*
 - void `setDate_envoie` (double value)
 - Cette méthode permet de définir la date à laquelle a été envoyé le message.*
 - `Utilisateur` * `getUser` () const
 - Cette méthode permet de récupérer l' Users qui est l'auteur du message.*
 - void `setUser` (`Utilisateur` *&value)
 - Cette méthode permet de définir l' `Utilisateur` qui l'auteur du message.*
 - `Ticket` * `getTicket` () const
 - Cette méthode permet de récupérer le `Ticket` dans lequel se trouve le message.*
 - void `setTicket` (`Ticket` *&value)
 - Cette méthode définis le `Ticket` dans lequel se trouve le message.*

3.7.1 Member Function Documentation

3.7.1.1 getDate_envoie()

```
double Message::getDate_envoie ( ) const
```

Cette méthode permet de récupérer la date à laquelle le message a été envoyé.

Returns

la date d'envoi du message

3.7.1.2 getId_message()

```
std::string Message::getId_message ( ) const
```

Cette méthode permet de récupérer l'identifiant du message.

Returns

l'identifiant du message

3.7.1.3 getTicket()

```
Ticket * Message::getTicket ( ) const
```

Cette méthode permet de récupérer le `Ticket` dans lequel se trouve le message.

Returns

la classe Users

3.7.1.4 getUser()

```
Utilisateur * Message::getUser ( ) const
```

Cette méthode permet de récupérer l' Users qui est l'auteur du message.

Parameters

<i>value</i>	le nouveau Users
--------------	------------------

3.7.1.5 setDate_envoie()

```
void Message::setDate_envoie (
    double value )
```

Cette méthode permet de définir la date à laquelle a été envoyé le message.

Parameters

<i>value</i>	la nouvelle date d'envoi
--------------	--------------------------

3.7.1.6 setId_message()

```
void Message::setId_message (
    const std::string & value )
```

Cette méthode permet de définir l'identifiant du message.

Parameters

<i>value</i>	Le nouvel identifiant du message
--------------	----------------------------------

3.7.1.7 setTicket()

```
void Message::setTicket (
    Ticket *& value )
```

Cette méthode définit le [Ticket](#) dans lequel se trouve le message.

Parameters

<i>value</i>	le nouveau Ticket
--------------	-----------------------------------

3.7.1.8 setUser()

```
void Message::setUser (
    Utilisateur * & value )
```

Cette méthode permet de définir l' **Utilisateur** qui l'auteur du message.

Parameters

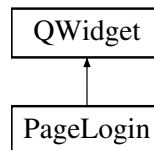
<i>value</i>	le nouveau Users
--------------	------------------

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

- include/message.h
- src/message.cpp

3.8 PageLogin Class Reference

Inheritance diagram for PageLogin:



Public Slots

- void **handle_validation** ()

Public Member Functions

- **PageLogin** (QWidget *parent=nullptr)

Protected Attributes

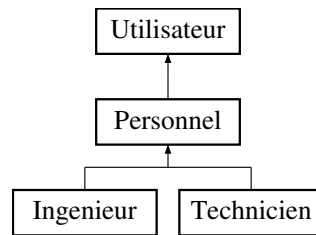
- QPushButton * **validation_boutton**
- QLineEdit * **id**
- QLineEdit * **mdp**

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

- include/pagelogin.h
- src/pagelogin.cpp

3.9 Personnel Class Reference

Inheritance diagram for Personnel:



Public Member Functions

- [Personnel](#) (std::string id_u, std::string n, std::string p, std::string motdepasse, std::string email)
La classe [Personnel](#) permet de décrire un membre du personnel dans l'application EasyTicket.
- virtual [~Personnel](#) ()=default
Déconstructeur virtuel pour rendre la classe [Personnel](#) abstraite.
- bool [estUnPersonnel](#) ()
Cette méthode permet de vérifier si le [Utilisateur](#) est un [Personnel](#).
- bool [estUnIngenieur](#) ()
Cette méthode permet de vérifier si le [Personnel](#) est un [Personnel](#).
- bool [estUnTechnicien](#) ()
Cette méthode permet de vérifier si un [Personnel](#) est un [Technicien](#).

3.9.1 Constructor & Destructor Documentation

3.9.1.1 Personnel()

```

Personnel::Personnel (
    std::string id_u,
    std::string n,
    std::string p,
    std::string motdepasse,
    std::string email )
  
```

La classe [Personnel](#) permet de décrire un membre du personnel dans l'application EasyTicket.

Parameters

<i>id_u</i>	l'identifiant du personnel
<i>n</i>	le nom du personnel
<i>p</i>	le prénom du personnel
<i>email</i>	l'email du personnel

3.9.2 Member Function Documentation

3.9.2.1 estUnIngenieur()

```
bool Personnel::estUnIngenieur ( )
```

Cette méthode permet de vérifier si le [Personnel](#) est un [Personnel](#).

Returns

false

3.9.2.2 estUnPersonnel()

```
bool Personnel::estUnPersonnel ( )
```

Cette méthode permet de vérifier si le [Utilisateur](#) est un [Personnel](#).

Returns

true

3.9.2.3 estUnTechnicien()

```
bool Personnel::estUnTechnicien ( )
```

Cette méthode permet de vérifier si un [Personnel](#) est un [Technicien](#).

Returns

false

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

- include/personnel.h
- src/personnel.cpp

3.10 Systeme Class Reference

Public Member Functions

- [Systeme](#) (std::string id_systeme, std::string nom)
La classe [Systeme](#) permet de décrire un système d'exploitation dans l'application EasyTicket.
- std::string [getId_systeme](#) () const
Cette méthode permet de récupérer l'identifiant du système.
- void [setId_systeme](#) (const std::string &value)
Cette méthode permet de définir l'identifiant du système.
- std::string [getNom](#) () const
Cette méthode permet de récupérer le nom du système.
- void [setNom](#) (const std::string &value)
Cette méthode permet de définir le nom du système.

3.10.1 Constructor & Destructor Documentation

3.10.1.1 Systeme()

```
Systeme::Systeme (
    std::string id_systeme,
    std::string nom )
```

La classe [Systeme](#) permet de décrire un système d'exploitation dans l'application EasyTicket.

Parameters

<i>id_systeme</i>	L'identifiant du système d'exploitation
<i>nom</i>	Le nom du système d'exploitation

3.10.2 Member Function Documentation

3.10.2.1 getId_systeme()

```
std::string Systeme::getId_systeme ( ) const
```

Cette méthode permet de récupérer l'identifiant du système.

Returns

l'identifiant du système

3.10.2.2 getNom()

```
std::string Systeme::getNom ( ) const
```

Cette méthode permet de récupérer le nom du système.

Returns

le nom du système

3.10.2.3 setId_systeme()

```
void Systeme::setId_systeme (
    const std::string & value )
```

Cette méthode permet de définir l'identifiant du système.

Parameters

<i>value</i>	l'identifiant du système
--------------	--------------------------

3.10.2.4 setNom()

```
void Systeme::setNom (
    const std::string & value )
```

Cette méthode permet de définir le nom du système.

Parameters

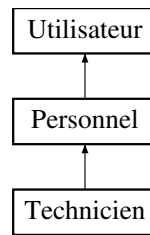
<i>value</i>	le nouveau nom du système
--------------	---------------------------

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

- include/systeme.h
- src/systeme.cpp

3.11 Technicien Class Reference

Inheritance diagram for Technicien:



Public Member Functions

- [Technicien](#) (std::string id_u, std::string n, std::string p, std::string motdepasse, std::string email, std::vector< [Categorie](#) * > liste_categories)

Cette classe permet de décrire un [Technicien](#) dans l'application EasyTicket.

- bool [estUnTechnicien](#) ()

Cette méthode permet de vérifier si un [Personnel](#) est un [Technicien](#).

3.11.1 Constructor & Destructor Documentation

3.11.1.1 Technicien()

```

Technicien::Technicien (
    std::string id_u,
    std::string n,
    std::string p,
    std::string motdepasse,
    std::string email,
    std::vector< Categorie * > liste_categories )
  
```

Cette classe permet de décrire un [Technicien](#) dans l'application EasyTicket.

Parameters

<i>id_u</i>	l'identifiant du technicien
<i>n</i>	le nom du technicien
<i>p</i>	le prénom du technicien
<i>motdepasse</i>	le mot de passe du technicien
<i>mail</i>	le mail du technicien
<i>liste_categories</i>	la liste des catégories que peut traiter le technicien

3.11.2 Member Function Documentation

3.11.2.1 estUnTechnicien()

```
bool Technicien::estUnTechnicien ( )
```

Cette méthode permet de vérifier si un [Personnel](#) est un [Technicien](#).

Returns

true

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

- include/technicien.h
- src/technicien.cpp

3.12 Ticket Class Reference

Public Member Functions

- [Ticket](#) (std::string id_ticket, double date_creation, [Systeme](#) systeme, [Logiciel](#) logiciel, [Client](#) auteur)
la classe [Ticket](#) permet de décrire un ticket dans l'application EasyTicket.
- std::string [getId_ticket](#) () const
Cette méthode permet de récupérer l'identifiant du [Ticket](#).
- void [setId_ticket](#) (const std::string &value)
Cette méthode permet de définir l'identifiant du [Ticket](#).
- double [getDate_creation](#) () const
Cette méthode permet de récupérer la date de création du ticket.
- void [setDate_creation](#) (double value)
Cette méthode permet de définir la date de création du [Ticket](#).
- double [getDate_fermeture](#) () const
Cette méthode permet de récupérer la date de fermeture du [Ticket](#).
- void [setDate_fermeture](#) (double value)
Cette méthode permet de définir la date de fermeture du [Ticket](#).
- [Systeme](#) [getSysteme](#) () const
Cette méthode permet de récupérer le système d'exploitation concerné par le [Ticket](#).
- void [setSysteme](#) (const [Systeme](#) &value)
Cette méthode permet de définir le [Systeme](#) concerné par le [Ticket](#).
- [Logiciel](#) [getLogiciel](#) () const
Cette méthode permet récupérer le [Logiciel](#) concerné par le [Ticket](#).
- void [setLogiciel](#) (const [Logiciel](#) &value)
Cette méthode permet définir le [Logiciel](#) concerné par le [Ticket](#).
- std::vector< [Message](#) * > [getListeMessages](#) () const
Cette méthode permet de récupérer la liste des [Message](#) concerné par le [Ticket](#).
- void [setListeMessages](#) (const std::vector< [Message](#) * > &value)
Cette méthode permet de définir la liste des @Message concerné par le [Ticket](#).
- void [setClient](#) ([Client](#) &value)
Cette méthode permet de définir le [Client](#) auteur du [Ticket](#).
- [Client](#) [getClient](#) ()
Cette méthode permet de récupérer le [Client](#) auteur du [Ticket](#).
- void [setPersonnel](#) ([Personnel](#) &value)
Cette méthode permet de définir le [Personnel](#) qui traite le [Ticket](#).
- [Personnel](#) [getPersonnel](#) ()
Cette méthode permet de récupérer le [Personnel](#) qui traite le [Ticket](#).

3.12.1 Constructor & Destructor Documentation

3.12.1.1 Ticket()

```
Ticket::Ticket (
    std::string id_ticket,
    double date_creation,
    Systeme systeme,
    Logiciel logiciel,
    Client auteur )
```

la classe [Ticket](#) permet de décrire un ticket dans l'application EasyTicket.

Parameters

<i>id_ticket</i>	l'identifiant du ticket
<i>date_creation</i>	la date de création du ticket
<i>systeme</i>	le Systeme concerné par le ticket
<i>logiciel</i>	le Logiciel concerné par le ticket
<i>auteur</i>	le Client auteur du ticket

3.12.2 Member Function Documentation

3.12.2.1 getClient()

```
Client Ticket::getClient ( )
```

Cette méthode permet de récupérer le [Client](#) auteur du [Ticket](#).

Returns

Le [Client](#) auteur du [Ticket](#)

3.12.2.2 getDate_creation()

```
double Ticket::getDate_creation ( ) const
```

Cette méthode permet de récupérer la date de création du ticket.

Returns

la date de création du ticket

3.12.2.3 getDate_fermeture()

```
double Ticket::getDate_fermeture ( ) const
```

Cette méthode permet de récupérer la date de fermeture du [Ticket](#).

Returns

la date de fermeture du ticket

3.12.2.4 getId_ticket()

```
std::string Ticket::getId_ticket ( ) const
```

Cette méthode permet de récupérer l'identifiant du [Ticket](#).

Returns

l'identifiant du ticket

3.12.2.5 getListeMessages()

```
std::vector< Message * > Ticket::getListeMessages ( ) const
```

Cette méthode permet de récupérer la liste des [Message](#) concerné par le [Ticket](#).

Returns

la liste de [Message](#) concerné par le ticket

3.12.2.6 getLogiciel()

```
Logiciel Ticket::getLogiciel ( ) const
```

Cette méthode permet récupérer le [Logiciel](#) concerné par le [Ticket](#).

Returns

le [Logiciel](#) concerné par le [Ticket](#)

3.12.2.7 `getPersonnel()`

```
Personnel Ticket::getPersonnel ( )
```

Cette méthode permet de récupérer le [Personnel](#) qui traite le [Ticket](#).

Returns

le Personel traitant le [Ticket](#)

3.12.2.8 `getSysteme()`

```
Systeme Ticket::getSysteme ( ) const
```

Cette méthode permet de récupérer le système d'exploitation concerné par le [Ticket](#).

Returns

la date de fermeture du ticket

3.12.2.9 `setClient()`

```
void Ticket::setClient (
    Client & value )
```

Cette méthode permet de définir le [Client](#) auteur du [Ticket](#).

Parameters

<i>value</i>	le Client auteur du Ticket
--------------	--

3.12.2.10 `setDate_creation()`

```
void Ticket::setDate_creation (
    double value )
```

Cette méthode permet de définir la date de création du [Ticket](#).

Parameters

<i>value</i>	la nouvelle date de création du Ticket
--------------	--

3.12.2.11 setDate_fermeture()

```
void Ticket::setDate_fermeture (
    double value )
```

Cette méthode permet de définir la date de fermeture du [Ticket](#).

Parameters

<i>value</i>	la nouvelle date de fermeture du Ticket
--------------	---

3.12.2.12 setId_ticket()

```
void Ticket::setId_ticket (
    const std::string & value )
```

Cette méthode permet de définir l'identifiant du [Ticket](#).

Parameters

<i>value</i>	le nouvel identifiant du ticket
--------------	---------------------------------

3.12.2.13 setListeMessages()

```
void Ticket::setListeMessages (
    const std::vector< Message * > & value )
```

Cette méthode permet de définir la liste des @Message concerné par le [Ticket](#).

Parameters

<i>value</i>	la nouvelle liste de Message
--------------	--

3.12.2.14 setLogiciel()

```
void Ticket::setLogiciel (
    const Logiciel & value )
```

Cette méthode permet définir le [Logiciel](#) concerné par le [Ticket](#).

Parameters

<i>value</i>	le nouveau Logiciel concerné par le Ticket
--------------	--

3.12.2.15 setPersonnel()

```
void Ticket::setPersonnel (  
    Personnel & value )
```

Cette méthode permet de définir le [Personnel](#) qui traite le [Ticket](#).

Parameters

<i>value</i>	le nouveau Personnel
--------------	--------------------------------------

3.12.2.16 setSysteme()

```
void Ticket::setSysteme (  
    const Systeme & value )
```

Cette méthode permet de définir le [Systeme](#) concerné par le [Ticket](#).

Parameters

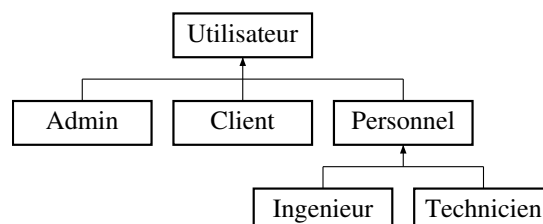
<i>value</i>	le nouveau Systeme concerné par le Ticket
--------------	---

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

- include/ticket.h
- src/ticket.cpp

3.13 Utilisateur Class Reference

Inheritance diagram for Utilisateur:



Public Member Functions

- [Utilisateur](#) (std::string id_u, std::string n, std::string p, std::string motdepasse, std::string email)
La classe [Utilisateur](#) est une classe abstraite qui permet de décrire un [Utilisateur](#) dans l'application EasyTicket.
- virtual [~Utilisateur](#) ()=default
Déconstructeur virtuel pour rendre la classe.
- std::string [getId_users](#) () const
Cette méthode permet de récupérer l'identifiant de l'utilisateur.
- void [setId_users](#) (const std::string &value)
Cette méthode permet de définir l'identifiant de l'utilisateur.
- std::string [getNom](#) () const
Cette méthode permet de récupérer le nom de l'utilisateur.
- void [setNom](#) (const std::string &value)
Cette méthode permet de définir le nom de l'utilisateur.
- std::string [getPrenom](#) () const
Cette méthode permet de récupérer le prénom de l'utilisateur.
- void [setPrenom](#) (const std::string &value)
Cette méthode permet de définir le prénom de l'utilisateur.
- std::string [getMdp](#) () const
Cette méthode permet de récupérer le mot de passe de l'utilisateur.
- void [setMdp](#) (const std::string &value)
Cette méthode permet de définir le mot de passe de l'utilisateur.
- std::string [getMail](#) () const
Cette méthode permet de récupérer le mail de l'utilisateur.
- void [setMail](#) (const std::string &value)
Cette méthode permet de définir le mail de l'utilisateur.
- bool [estUnClient](#) ()
Cette méthode permet de vérifier si un [Utilisateur](#) est un [Client](#).
- bool [estUnPersonnel](#) ()
Cette méthode permet de vérifier si le [Utilisateur](#) est un [Personnel](#).
- bool [estUnAdmin](#) ()
Cette méthode permet de vérifier si un [Utilisateur](#) est un [Admin](#).

3.13.1 Constructor & Destructor Documentation

3.13.1.1 Utilisateur()

```
Utilisateur::Utilisateur (
    std::string id_u,
    std::string n,
    std::string p,
    std::string motdepasse,
    std::string email )
```

La classe [Utilisateur](#) est une classe abstraite qui permet de décrire un [Utilisateur](#) dans l'application EasyTicket.

Parameters

<i>id_u</i>	L'identifiant de l'utilisateur
<i>n</i>	Le nom de l'utilisateur
<i>p</i>	Le prénom de l'utilisateur
<i>motdepasse</i>	Le mot de passe de l'utilisateur
<i>email</i>	Le mail de l'utilisateur

3.13.2 Member Function Documentation

3.13.2.1 `estUnAdmin()`

```
bool Utilisateur::estUnAdmin ( )
```

Cette méthode permet de vérifier si un [Utilisateur](#) est un [Admin](#).

Returns

false

3.13.2.2 `estUnClient()`

```
bool Utilisateur::estUnClient ( )
```

Cette méthode permet de vérifier si un [Utilisateur](#) est un [Client](#).

Returns

false

3.13.2.3 `estUnPersonnel()`

```
bool Utilisateur::estUnPersonnel ( )
```

Cette méthode permet de vérifier si le [Utilisateur](#) est un [Personnel](#).

Returns

false

3.13.2.4 getId_users()

```
std::string Utilisateur::getId_users ( ) const
```

Cette méthode permet de récupérer l'identifiant de l'utilisateur.

Returns

L'identifiant de l'utilisateur.

3.13.2.5 getMail()

```
std::string Utilisateur::getMail ( ) const
```

Cette méthode permet de récupérer le mail de l'utilisateur.

Returns

Le mail de l'utilisateur.

3.13.2.6 getMdp()

```
std::string Utilisateur::getMdp ( ) const
```

Cette méthode permet de récupérer le mot de passe de l'utilisateur.

Returns

Le mot de passe de l'utilisateur.

3.13.2.7 getNom()

```
std::string Utilisateur::getNom ( ) const
```

Cette méthode permet de récupérer le nom de l'utilisateur.

Returns

Le nom de l'utilisateur.

3.13.2.8 getPrenom()

```
std::string Utilisateur::getPrenom ( ) const
```

Cette méthode permet de récupérer le prénom de l'utilisateur.

Returns

Le prénom de l'utilisateur.

3.13.2.9 setId_users()

```
void Utilisateur::setId_users (
    const std::string & value )
```

Cette méthode permet de définir l'identifiant de l'utilisateur.

Parameters

<i>value</i>	le nouvel identifiant de l'utilisateur.
--------------	---

3.13.2.10 setMail()

```
void Utilisateur::setMail (
    const std::string & value )
```

Cette méthode permet de définir le mail de l'utilisateur.

Parameters

<i>value</i>	le mail
--------------	---------

3.13.2.11 setMdp()

```
void Utilisateur::setMdp (
    const std::string & value )
```

Cette méthode permet de définir le mot de passe de l'utilisateur.

Parameters

<i>value</i>	Le nouveau mot de passe de l'utilisateur.
--------------	---

3.13.2.12 setNom()

```
void Utilisateur::setNom (
    const std::string & value )
```

Cette méthode permet de définir le nom de l'utilisateur.

Parameters

<i>value</i>	le nouveau nom de l'utilisateur.
--------------	----------------------------------

3.13.2.13 setPrenom()

```
void Utilisateur::setPrenom (
    const std::string & value )
```

Cette méthode permet de définir le prénom de l'utilisateur.

Parameters

<i>value</i>	Le nouveau prénom de l'utilisateur.
--------------	-------------------------------------

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

- include/utilisateur.h
- src/utilisateur.cpp

Index

- Admin, [5](#)
 - Admin, [5](#)
 - estUnAdmin, [6](#)
- Categorie, [6](#)
 - Categorie, [6](#)
- Client, [7](#)
 - Client, [7](#)
 - estUnClient, [7](#)
- estUnAdmin
 - Admin, [6](#)
 - Utilisateur, [26](#)
- estUnClient
 - Client, [7](#)
 - Utilisateur, [26](#)
- estUnIngenieur
 - Ingenieur, [9](#)
 - Personnel, [15](#)
- estUnPersonnel
 - Personnel, [15](#)
 - Utilisateur, [26](#)
- estUnTechnicien
 - Personnel, [15](#)
 - Technicien, [18](#)
- getClient
 - Ticket, [20](#)
- getDate_creation
 - Ticket, [20](#)
- getDate_envoie
 - Message, [11](#)
- getDate_fermeture
 - Ticket, [20](#)
- getId_message
 - Message, [11](#)
- getId_systeme
 - Systeme, [16](#)
- getId_ticket
 - Ticket, [21](#)
- getId_users
 - Utilisateur, [26](#)
- getListeMessages
 - Ticket, [21](#)
- getLogiciel
 - Ticket, [21](#)
- getMail
 - Utilisateur, [27](#)
- getMdp
 - Utilisateur, [27](#)

- getNom
 - Systeme, [16](#)
 - Utilisateur, [27](#)
- getPersonnel
 - Ticket, [21](#)
- getPrenom
 - Utilisateur, [27](#)
- getSysteme
 - Ticket, [22](#)
- getTicket
 - Message, [11](#)
- getUser
 - Message, [11](#)
- Ingenieur, [8](#)
 - estUnIngenieur, [9](#)
 - Ingenieur, [8](#)
- Logiciel, [9](#)
 - Logiciel, [9](#)
- MainWindow, [10](#)
- Message, [10](#)
 - getDate_envoie, [11](#)
 - getId_message, [11](#)
 - getTicket, [11](#)
 - getUser, [11](#)
 - getDate_envoie, [12](#)
 - setId_message, [12](#)
 - setTicket, [12](#)
 - setUser, [12](#)
- PageLogin, [13](#)
- Personnel, [14](#)
 - estUnIngenieur, [15](#)
 - estUnPersonnel, [15](#)
 - estUnTechnicien, [15](#)
 - Personnel, [14](#)
- setClient
 - Ticket, [22](#)
- setDate_creation
 - Ticket, [22](#)
- setDate_envoie
 - Message, [12](#)
- setDate_fermeture
 - Ticket, [23](#)
- setId_message
 - Message, [12](#)
- setId_systeme
 - Systeme, [17](#)

setId_ticket
 Ticket, 23
setId_users
 Utilisateur, 28
setListeMessages
 Ticket, 23
setLogiciel
 Ticket, 23
setMail
 Utilisateur, 28
setMdp
 Utilisateur, 28
setNom
 Systeme, 17
 Utilisateur, 29
setPersonnel
 Ticket, 24
setPrenom
 Utilisateur, 29
setSysteme
 Ticket, 24
setTicket
 Message, 12
setUser
 Message, 12
Systeme, 16
 getId_systeme, 16
 getNom, 16
 setId_systeme, 17
 setNom, 17
 Systeme, 16

Technicien, 17
 estUnTechnicien, 18
 Technicien, 18
Ticket, 19
 getClient, 20
 getDate_creation, 20
 getDate_fermeture, 20
 getId_ticket, 21
 getListeMessages, 21
 getLogiciel, 21
 getPersonnel, 21
 getSysteme, 22
 setClient, 22
 setDate_creation, 22
 setDate_fermeture, 23
 setId_ticket, 23
 setListeMessages, 23
 setLogiciel, 23
 setPersonnel, 24
 setSysteme, 24
 Ticket, 20

Utilisateur, 24
 estUnAdmin, 26
 estUnClient, 26
 estUnPersonnel, 26
 getId_users, 26
 getMail, 27
 getMdp, 27
 getNom, 27
 getPrenom, 27
 setId_users, 28
 setMail, 28
 setMdp, 28
 setNom, 29
 setPrenom, 29
 Utilisateur, 25