

My Project

Generated by Doxygen 1.9.8

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 Angajat Class Reference	7
4.2 AngajatFactory Class Reference	8
4.3 AngajatiManager Class Reference	9
4.4 CerereFactory Class Reference	9
4.5 CerereManager Class Reference	9
4.6 CerereReparatie Class Reference	10
4.7 CNPValidator Class Reference	10
4.8 Data Class Reference	11
4.9 Electrocasnic Class Reference	11
4.10 ElectrocasniceManager Class Reference	12
4.11 ElectrocasnicFactory Class Reference	13
4.12 Frigider Class Reference	13
4.12.1 Member Function Documentation	14
4.12.1.1 afisare()	14
4.12.1.2 cloneaza()	14
4.13 MasinaDeSpalat Class Reference	15
4.13.1 Member Function Documentation	16
4.13.1.1 afisare()	16
4.13.1.2 cloneaza()	16
4.14 Meniu Class Reference	16
4.15 Receptioner Class Reference	17
4.15.1 Member Function Documentation	18
4.15.1.1 afisare()	18
4.15.1.2 calculeazaSalariu()	18
4.15.1.3 getTipAngajat()	19
4.16 ServiceManager Class Reference	19
4.17 Supervizor Class Reference	19
4.17.1 Member Function Documentation	21
4.17.1.1 afisare()	21
4.17.1.2 calculeazaSalariu()	21
4.17.1.3 getTipAngajat()	21
4.18 Tehnician Class Reference	21
4.18.1 Member Function Documentation	23

4.18.1.1 afisare()	23
4.18.1.2 calculeazaSalariu()	23
4.18.1.3 getTipAngajat()	23
4.19 TV Class Reference	24
4.19.1 Member Function Documentation	25
4.19.1.1 afisare()	25
4.19.1.2 cloneaza()	25
5 File Documentation	27
5.1 Angajat.h	27
5.2 AngajatFactory.h	28
5.3 AngajatiManager.h	28
5.4 CerereFactory.h	29
5.5 CerereManager.h	29
5.6 CerereReparatie.h	29
5.7 CNPValidator.h	30
5.8 Data.h	31
5.9 Electrocasnic.h	31
5.10 ElectrocasniceManager.h	31
5.11 ElectrocasnicFactory.h	32
5.12 Frigider.h	32
5.13 MasinaDeSpalat.h	33
5.14 Meniu.h	33
5.15 Receptioner.h	33
5.16 ServiceManager.h	33
5.17 Supervizor.h	34
5.18 Tehnician.h	34
5.19 TV.h	35
Index	37

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Angajat	7
Receptioner	17
Supervizor	19
Tehnician	21
AngajatFactory	8
AngajatiManager	9
CerereFactory	9
CerereManager	9
CerereReparatie	10
CNPValidator	10
Data	11
Electrocasnic	11
Frigider	13
MasinaDeSpalat	15
TV	24
ElectrocasniceManager	12
ElectrocasnicFactory	13
Meniu	16
ServiceManager	19

Chapter 2

Class Index

2.1 Class List

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

Angajat	7
AngajatFactory	8
AngajatiManager	9
CerereFactory	9
CerereManager	9
CerereReparatie	10
CNPValidator	10
Data	11
Electrocasnic	11
ElectrocashniceManager	12
ElectrocasnicFactory	13
Frigider	13
MasinaDeSpalat	15
Meniu	16
Receptioner	17
ServiceManager	19
Supervizor	19
Tehnician	21
TV	24

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

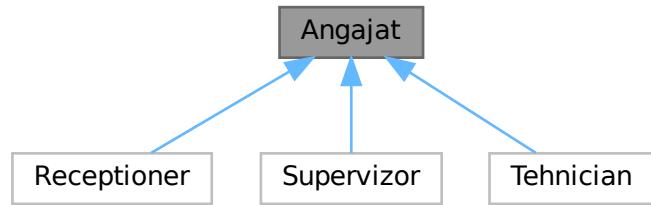
Angajat.h	27
AngajatFactory.h	28
AngajatiManager.h	28
CerereFactory.h	29
CerereManager.h	29
CerereReparatie.h	29
CNPValidator.h	30
Data.h	31
Electrocasnic.h	31
ElectrocashniceManager.h	31
ElectrocasnicFactory.h	32
Frigider.h	32
MasinaDeSpalat.h	33
Meniu.h	33
Receptioner.h	33
ServiceManager.h	33
Supervizor.h	34
Tehnician.h	34
TV.h	35

Chapter 4

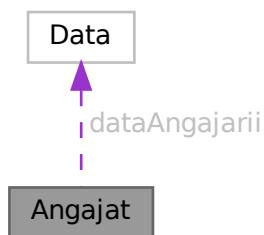
Class Documentation

4.1 Angajat Class Reference

Inheritance diagram for Angajat:



Collaboration diagram for Angajat:



Public Member Functions

- **Angajat** (const std::string &, const std::string &, const std::string &, const [Data](#) &, const std::string &)
- int **getId** () const
- std::string **getNum** () const
- std::string **getPrenume** () const
- std::string **getCNP** () const
- [Data](#) **getDataAngajarii** () const
- std::string **getOrasDomiciliu** () const
- void **setNum** (const std::string &)
- void **setPrenume** (const std::string &)
- void **validareNum** (const std::string &, const std::string &) const
- virtual double **calculeazaSalariu** () const =0
- virtual void **afisare** (std::ostream &) const
- virtual std::string **getTipAngajat** () const =0
- double **calculeazaBonusFidelitate** () const
- bool **primestePrimaTransport** () const

Protected Attributes

- const int **id**
- std::string **nume**
- std::string **prenume**
- std::string **cnp**
- [Data](#) **dataAngajarii**
- std::string **orasDomiciliu**

Static Protected Attributes

- static int **nextID** =1000
- static constexpr double **SALARIU_BAZA** =4000.0
- static constexpr double **BONUS_DE_FIDELITARE_PROCENT** =5.0
- static constexpr int **ANI_FIDELITATE** =3
- static constexpr double **PRIMA_TRANS** =400.0

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

- Angajat.h
- Angajat.cpp

4.2 AngajatFactory Class Reference

Static Public Member Functions

- static std::unique_ptr< [Angajat](#) > **creeazaAngajat** (const std::string &tip, const std::string &nume, const std::string &prenume, const std::string &cnp, const [Data](#) &dataAngajarii, const std::string &orasDomiciliu, const std::map< std::string, std::set< std::string > > &specializari={})
- static std::unique_ptr< [Angajat](#) > **creeazaDinCSV** (const std::string &linie)

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

- AngajatFactory.h
- AngajatFactory.cpp

4.3 AngajatiManager Class Reference

Public Member Functions

- void **incarcaDinCSV** (const std::string &numeFisier)
- **Tehnician * gasesteTehnicianDisponibil** (const std::string &tipAparat, const std::string &marcaAparat)
- void **adaugaAngajat** (std::unique_ptr< **Angajat** > a)
- bool **stergeAngajat** (const std::string &cnp)
- bool **modificaNume** (const std::string &cnp, const std::string &numeNou, const std::string &prenumeNou)
- **Angajat * cautaDupaCNP** (const std::string &cnp)
- **Angajat * cautaDupaID** (int id)
- bool **verificaPersonalMinim** () const
- bool **adaugaAngajatManual** (std::string tip, std::string nume, std::string prenume, std::string cnp, std::string data, std::string oras)
- void **afiseazaToti** () const
- std::vector< **Angajat** * > **getTop3Salarii** () const
- void **genereazaRaportTop3Salarii** (const std::string &numeFisier) const
- **Tehnician * getTehnicianCeaMaiLungaReparatie** () const

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

- AngajatiManager.h
- AngajatiManager.cpp

4.4 CerereFactory Class Reference

Static Public Member Functions

- static std::unique_ptr< **CerereReparatie** > **creeazaDinCSV** (const std::string &linie)

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

- CerereFactory.h
- CerereFactory.cpp

4.5 CerereManager Class Reference

Public Member Functions

- void **incarcaDinCSV** (const std::string &fisier, **ElectrocasniceManager** &catalog)
- void **adaugaCerere** (std::unique_ptr< **CerereReparatie** > c)
- **CerereReparatie * getUrmatoareaInAsteptare** ()
- void **popAsteptare** ()
- void **puneInapoilnAsteptare** (**CerereReparatie** *c)
- bool **areCereriInAsteptare** () const
- void **repartizeazaCerere** (**CerereReparatie** *c, int idTehnician)
- void **mutaInFinalizate** (**CerereReparatie** *c)
- std::vector< **CerereReparatie** * > & **getCereriInLucru** ()
- bool **areCereriActive** () const
- void **afiseazaStatisticiCurente** () const
- std::vector< int > **getIdsInAsteptare** () const
- void **genereazaRaportCereriInAsteptare** (const std::string &numeFisier) const
- void **afiseazalstoricReparatii** () const

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

- CerereManager.h
- CerereManager.cpp

4.6 CerereReparatie Class Reference

Public Member Functions

- `CerereReparatie (std::unique_ptr< Electrocasicnic > _aparat, const time_t _timestamp, int _nivelComplexitate)`
- `CerereReparatie (const CerereReparatie &)`
- `CerereReparatie & operator= (const CerereReparatie &)`
- `int getId () const`
- `int getNivelComplexitate () const`
- `const Electrocasicnic * getAparat () const`
- `time_t getTimeStamp () const`
- `int getDurataEstimata () const`
- `int getDurataRamasa () const`
- `double getPretReparatie () const`
- `StareCerere getStare () const`
- `int getIdTehnician () const`
- `void setStare (const StareCerere)`
- `void setIdTehnician (int)`
- `void proceseaza ()`
- `bool esteFinalizata () const`
- `bool esteValida () const`
- `void afisare (std::ostream &) const`
- `std::string getTimeStampString () const`
- `bool operator< (const CerereReparatie &) const`

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

- CerereReparatie.h
- CerereReparatie.cpp

4.7 CNPValidator Class Reference

Static Public Member Functions

- static bool `esteValid (const std::string &cnp)`
- static `Data getDataMasterii (const std::string &cnp)`
- static char `getSex (const std::string &cnp)`

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

- CNPValidator.h
- CNPValidator.cpp

4.8 Data Class Reference

Public Member Functions

- **Data** (int, int, int)
- **Data** (const std::string &)
- bool **esteValida** () const
- int **getVarsta** () const
- int **getZi** () const
- int **getLuna** () const
- int **getAn** () const
- bool **operator<** (const **Data** &) const
- bool **operator>** (const **Data** &) const
- bool **operator==** (const **Data** &) const

Static Public Member Functions

- static **Data** **dataCurenta** ()

Friends

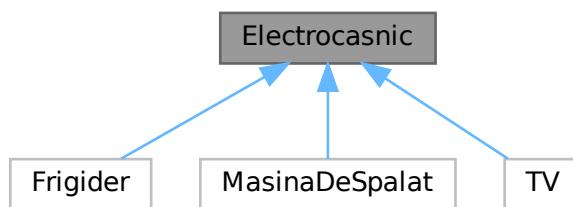
- std::ostream & **operator<<** (std::ostream &, const **Data** &)

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

- Data.h
- Data.cpp

4.9 Electrocasicnic Class Reference

Inheritance diagram for Electrocasicnic:



Public Member Functions

- **Electrocasnic** (const std::string &, const std::string &, const std::string &, int, double)
- std::string **getTip** () const
- std::string **getMarca** () const
- std::string **getModel** () const
- int **getAnFabricatie** () const
- double **getPretCatalog** () const
- int **getVechime** () const
- virtual void **afisare** (std::ostream &) const =0
- virtual std::unique_ptr< **Electrocasnic** > **cloneaza** () const =0
- bool **operator==** (const **Electrocasnic** &altul) const

Protected Attributes

- std::string **tip**
- std::string **marca**
- std::string **model**
- int **anFabricatie**
- double **pretCatalog**

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

- Electrocasnic.h
- Electrocasnic.cpp

4.10 ElectrocasniceManager Class Reference

Public Member Functions

- void **incarcaDinCSV** (const std::string &numeFisier)
- void **adaugaInCatalog** (std::unique_ptr< **Electrocasnic** > e)
- bool **existaModel** (const std::string &tip, const std::string &marca, const std::string &model) const
- const **Electrocasnic** * **getDetaliiModel** (const std::string &tip, const std::string &marca, const std::string &model) const
- void **adaugaModelManual** (std::string tip, std::string marca, std::string model, int an, double pret)
- bool **stergeModel** (std::string marca, std::string model)
- void **inregistreazaCerereInvalida** (const std::string &tip, const std::string &marca, const std::string &model)
- void **afiseazaAparateNereparabile** () const
- void **afiseazaCatalog** () const

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

- ElectrocasniceManager.h
- ElectrocasniceManager.cpp

4.11 ElectrocasnicFactory Class Reference

Static Public Member Functions

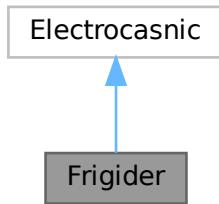
- static std::unique_ptr< [Electrocasnic](#) > **creeazaDinCSV** (const std::string &linie)

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

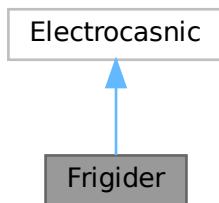
- [ElectrocasnicFactory.h](#)
- [ElectrocasnicFactory.cpp](#)

4.12 Frigider Class Reference

Inheritance diagram for Frigider:



Collaboration diagram for Frigider:



Public Member Functions

- **Frigider** (const std::string &, const std::string &, int, double, bool)
- bool **getAreCongelator** () const
- void **afisare** (std::ostream &) const override
- std::unique_ptr< [Electrocasnic](#) > **cloneaza** () const override

Public Member Functions inherited from [Electrocasicnic](#)

- **Electrocasicnic** (const std::string &, const std::string &, const std::string &, int, double)
- std::string **getTip** () const
- std::string **getMarca** () const
- std::string **getModel** () const
- int **getAnFabricatie** () const
- double **getPretCatalog** () const
- int **getVechime** () const
- bool **operator==** (const [Electrocasicnic](#) &altul) const

Additional Inherited Members

Protected Attributes inherited from [Electrocasicnic](#)

- std::string **tip**
- std::string **marca**
- std::string **model**
- int **anFabricatie**
- double **pretCatalog**

4.12.1 Member Function Documentation

4.12.1.1 afisare()

```
void Frigider::afisare (
    std::ostream & dev ) const [override], [virtual]
```

Implements [Electrocasicnic](#).

4.12.1.2 cloneaza()

```
unique_ptr< Electrocasicnic > Frigider::cloneaza ( ) const [override], [virtual]
```

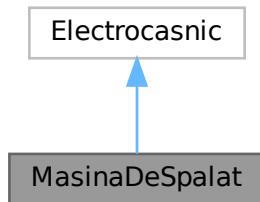
Implements [Electrocasicnic](#).

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

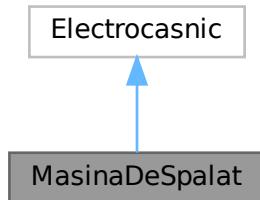
- Frigider.h
- Frigider.cpp

4.13 MasinaDeSpalat Class Reference

Inheritance diagram for MasinaDeSpalat:



Collaboration diagram for MasinaDeSpalat:



Public Member Functions

- **MasinaDeSpalat** (const std::string &, const std::string &, int, double, double)
- double **getCapacitate** () const
- void **afisare** (std::ostream &) const override
- std::unique_ptr< [Electrocasicnic](#) > **cloneaza** () const override

Public Member Functions inherited from [Electrocasicnic](#)

- **Electrocasicnic** (const std::string &, const std::string &, const std::string &, int, double)
- std::string **getTip** () const
- std::string **getMarca** () const
- std::string **getModel** () const
- int **getAnFabricatie** () const
- double **getPretCatalog** () const
- int **getVechime** () const
- bool **operator==** (const [Electrocasicnic](#) &altul) const

Additional Inherited Members

Protected Attributes inherited from [Electrocasicnic](#)

- std::string **tip**
- std::string **marca**
- std::string **model**
- int **anFabricatie**
- double **pretCatalog**

4.13.1 Member Function Documentation

4.13.1.1 **afisare()**

```
void MasinaDeSpalat::afisare (
    std::ostream & ) const [override], [virtual]
```

Implements [Electrocasicnic](#).

4.13.1.2 **cloneaza()**

```
unique_ptr< Electrocasicnic > MasinaDeSpalat::cloneaza () const [override], [virtual]
```

Implements [Electrocasicnic](#).

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

- [MasinaDeSpalat.h](#)
- [MasinaDeSpalat.cpp](#)

4.14 Meniu Class Reference

Public Member Functions

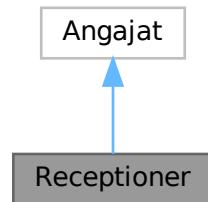
- void **ruleazaAplicatie ()**

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

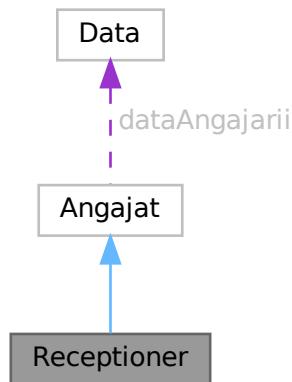
- [Meniu.h](#)
- [Meniu.cpp](#)

4.15 Receptioner Class Reference

Inheritance diagram for Receptioner:



Collaboration diagram for Receptioner:



Public Member Functions

- **Receptioner** (const std::string &, const std::string &, const std::string &, const [Data](#) &, const std::string &)
- void **adaugaCerere** (int)
- const std::vector< int > & **getCereri** () const
- double **calculeazaSalariu** () const override
- void **afisare** (std::ostream &) const override
- std::string **getTipAngajat** () const override

Public Member Functions inherited from [Angajat](#)

- **Angajat** (const std::string &, const std::string &, const std::string &, const [Data](#) &, const std::string &)
- int **getId** () const
- std::string **getNume** () const
- std::string **getPrenume** () const
- std::string **getCNP** () const
- [Data](#) **getDataAngajarii** () const
- std::string **getOrasDomiciliu** () const
- void **setNume** (const std::string &)
- void **setPrenume** (const std::string &)
- void **validareNume** (const std::string &, const std::string &) const
- double **calculeazaBonusFidelitate** () const
- bool **primestePrimaTransport** () const

Additional Inherited Members

Protected Attributes inherited from [Angajat](#)

- const int **id**
- std::string **nume**
- std::string **prenume**
- std::string **cnp**
- [Data](#) **dataAngajarii**
- std::string **orasDomiciliu**

Static Protected Attributes inherited from [Angajat](#)

- static int **nextID** =1000
- static constexpr double **SALARIU_BAZA** =4000.0
- static constexpr double **BONUS_DE_FIDELITARE_PROCENT** =5.0
- static constexpr int **ANI_FIDELITATE** =3
- static constexpr double **PRIMA_TRANS** =400.0

4.15.1 Member Function Documentation

4.15.1.1 [afisare\(\)](#)

```
void Receptioner::afisare (
    std::ostream & ) const [override], [virtual]
```

Reimplemented from [Angajat](#).

4.15.1.2 [calculeazaSalariu\(\)](#)

```
double Receptioner::calculeazaSalariu ( ) const [override], [virtual]
```

Implements [Angajat](#).

4.15.1.3 getTipAngajat()

```
string Receptioner::getTipAngajat ( ) const [override], [virtual]
```

Implements [Angajat](#).

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

- Receptioner.h
- Receptioner.cpp

4.16 ServiceManager Class Reference

Public Member Functions

- [ServiceManager \(const ServiceManager &\)=delete](#)
- [ServiceManager & operator= \(const ServiceManager &\)=delete](#)
- void [incarcaDate \(const std::string &fAng, const std::string &fEl, const std::string &fCer\)](#)
- void [runSimulare \(\)](#)
- [AngajatiManager & getAngajatiManager \(\)](#)
- [ElectrocasniceManager & getElectrocasniceManager \(\)](#)
- [CerereManager & getCerereManager \(\)](#)

Static Public Member Functions

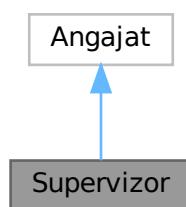
- static [ServiceManager & getInstance \(\)](#)

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

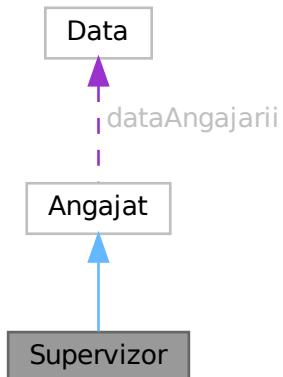
- ServiceManager.h
- ServiceManager.cpp

4.17 Supervizor Class Reference

Inheritance diagram for Supervizor:



Collaboration diagram for Supervizor:



Public Member Functions

- **Supervizor** (const std::string &, const std::string &, const std::string &, const [Data](#) &, const std::string &)
- double [calculeazaSalariu](#) () const override
- void [afisare](#) (std::ostream &) const override
- std::string [getTipAngajat](#) () const override

Public Member Functions inherited from [Angajat](#)

- **Angajat** (const std::string &, const std::string &, const std::string &, const [Data](#) &, const std::string &)
- int [getId](#) () const
- std::string [getNume](#) () const
- std::string [getPrenume](#) () const
- std::string [getCNP](#) () const
- [Data](#) [getDataAngajarii](#) () const
- std::string [getOrasDomiciliu](#) () const
- void [setNume](#) (const std::string &)
- void [setPrenume](#) (const std::string &)
- void [validareNume](#) (const std::string &, const std::string &) const
- double [calculeazaBonusFidelitate](#) () const
- bool [primestePrimaTransport](#) () const

Additional Inherited Members

Protected Attributes inherited from [Angajat](#)

- const int **id**
- std::string **nume**
- std::string **prenume**
- std::string **cnp**
- [Data](#) [dataAngajarii](#)
- std::string **orasDomiciliu**

Static Protected Attributes inherited from [Angajat](#)

- static int **nextID** =1000
- static constexpr double **SALARIU_BAZA** =4000.0
- static constexpr double **BONUS_DE_FIDELITARE_PROCENT** =5.0
- static constexpr int **ANI_FIDELITATE** =3
- static constexpr double **PRIMA_TRANS** =400.0

4.17.1 Member Function Documentation

4.17.1.1 [afisare\(\)](#)

```
void Supervizor::afisare (
    std::ostream & ) const [override], [virtual]
```

Reimplemented from [Angajat](#).

4.17.1.2 [calculeazaSalariu\(\)](#)

```
double Supervizor::calculeazaSalariu ( ) const [override], [virtual]
```

Implements [Angajat](#).

4.17.1.3 [getTipAngajat\(\)](#)

```
string Supervizor::getTipAngajat ( ) const [override], [virtual]
```

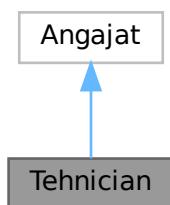
Implements [Angajat](#).

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

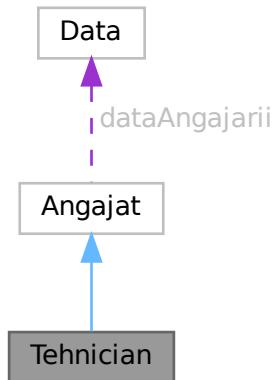
- Supervizor.h
- Supervizor.cpp

4.18 Tehnician Class Reference

Inheritance diagram for Tehnician:



Collaboration diagram for Tehnician:



Public Member Functions

- **Tehnician** (const std::string &_nume, const std::string &_prenume, const std::string &_cnp, const **Data** &←
dataAngajarii, const std::string &_orasDomiciliu)
- void **adaugaSpecializare** (const std::string &tip, const std::string &marca)
- bool **areSpecializare** (const std::string &tip, const std::string &marca) const
- bool **poatePrimiCerere** () const
- void **adaugaldCerereActivă** (int id)
- void **finalizareCerere** (int idCerere, double valoareReparatie, int durataEfectiva)
- double **getDurataTotalaLucrata** () const
- int **getNrCereriActive** () const
- double **calculeazaSalariu** () const override
- void **afisare** (std::ostream &) const override
- std::string **getTipAngajat** () const override

Public Member Functions inherited from **Angajat**

- **Angajat** (const std::string &, const std::string &, const std::string &, const **Data** &, const std::string &)
- int **getId** () const
- std::string **getNumă** () const
- std::string **getPrenume** () const
- std::string **getCNP** () const
- **Data** **getDataAngajarii** () const
- std::string **getOrasDomiciliu** () const
- void **setNumă** (const std::string &)
- void **setPrenume** (const std::string &)
- void **validareNumă** (const std::string &, const std::string &) const
- double **calculeazaBonusFidelitate** () const
- bool **primeștePrimaTransport** () const

Additional Inherited Members

Protected Attributes inherited from [Angajat](#)

- const int **id**
- std::string **nume**
- std::string **prenume**
- std::string **cnp**
- [Data](#) **dataAngajarii**
- std::string **orasDomiciliu**

Static Protected Attributes inherited from [Angajat](#)

- static int **nextID** =1000
- static constexpr double **SALARIU_BAZA** =4000.0
- static constexpr double **BONUS_DE_FIDELITARE_PROCENT** =5.0
- static constexpr int **ANI_FIDELITATE** =3
- static constexpr double **PRIMA_TRANS** =400.0

4.18.1 Member Function Documentation

4.18.1.1 [afisare\(\)](#)

```
void Tehnician::afisare (
    std::ostream & ) const [override], [virtual]
```

Reimplemented from [Angajat](#).

4.18.1.2 [calculeazaSalariu\(\)](#)

```
double Tehnician::calculeazaSalariu () const [override], [virtual]
```

Implements [Angajat](#).

4.18.1.3 [getTipAngajat\(\)](#)

```
string Tehnician::getTipAngajat () const [override], [virtual]
```

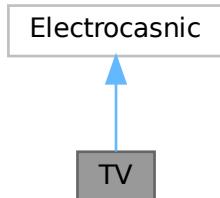
Implements [Angajat](#).

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

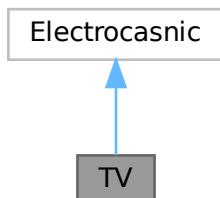
- Tehnician.h
- Tehnician.cpp

4.19 TV Class Reference

Inheritance diagram for TV:



Collaboration diagram for TV:



Public Member Functions

- **TV** (const std::string &, const std::string &, int, double, double, bool)
- double **getDiagonala** () const
- bool **getEstelnCm** () const
- void **afisare** (std::ostream &) const override
- std::unique_ptr<[Electrocasicnic](#)> **cloneaza** () const override

Public Member Functions inherited from [Electrocasicnic](#)

- **Electrocasicnic** (const std::string &, const std::string &, const std::string &, int, double)
- std::string **getTip** () const
- std::string **getMarca** () const
- std::string **getModel** () const
- int **getAnFabricatie** () const
- double **getPretCatalog** () const
- int **getVechime** () const
- bool **operator==** (const [Electrocasicnic](#) &altul) const

Additional Inherited Members

Protected Attributes inherited from [Electrocasnic](#)

- std::string **tip**
- std::string **marca**
- std::string **model**
- int **anFabricatie**
- double **pretCatalog**

4.19.1 Member Function Documentation

4.19.1.1 [afisare\(\)](#)

```
void TV::afisare ( std::ostream & ) const [override], [virtual]
```

Implements [Electrocasnic](#).

4.19.1.2 [cloneaza\(\)](#)

```
unique_ptr< Electrocasnic > TV::cloneaza ( ) const [override], [virtual]
```

Implements [Electrocasnic](#).

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

- TV.h
- TV.cpp

Chapter 5

File Documentation

5.1 Angajat.h

```
00001 #pragma once
00002 #include <string>
00003 #include <iostream>
00004 #include "Data.h"
00005
00006 class Angajat{
00007
00008 protected:
00009
00010     //TODO : gandeste te la o denumire mai buna pentru id urile de la angajat si cerere
00011     static int nextID;
00012     const int id;
00013     std::string nume, prenume, cnp;
00014     Data dataAngajarii;
00015     std::string orasDomiciliu;
00016
00017     static constexpr double SALARIU_BAZA=4000.0;
00018     static constexpr double BONUS_DE_FIDELITARE_PROCENT=5.0;
00019     static constexpr int ANI_FIDELITATE=3;
00020     static constexpr double PRIMA_TRANS=400.0;
00021
00022 public:
00023
00024     Angajat():id(0){}
00025     Angajat(const std::string& ,const std::string& , const std::string& , const Data& , const
00026     std::string& );
00027     virtual ~Angajat()=default;
00028
00029     // getteri
00030
00031     int getId() const;
00032     std::string getNume() const;
00033     std::string getPrenume() const;
00034     std::string getCNP() const;
00035     Data getDataAngajarii() const;
00036     std::string getOrasDomiciliu() const;
00037
00038     //setteri
00039
00040     void setNume(const std::string& );
00041     void setPrenume(const std::string& );
00042
00043     //validare
00044
00045     void validareNume(const std::string& , const std::string& ) const;
00046
00047     void calculSalariu-- VIRTUAL PURE
00048     virtual double calculeazaSalariu() const=0;
00049     virtual void afisare(std::ostream& ) const;
00050
00051     //tip angajat-- VIRTUAL PURE
00052     virtual std::string getTipAngajat() const=0;
00053
00054
00055
00056
00057
```

```

00058 // auxiliare
00059 double calculeazaBonusFidelitate() const;
00060 bool primestePrimaTransport() const;
00061
00062 };

```

5.2 AngajatFactory.h

```

00001
00002
00003 #pragma once
00004 #include <string>
00005 #include <memory>
00006 #include <vector>
00007 #include <map>
00008 #include <set>
00009 #include "Angajat.h"
00010
00011 class AngajatFactory {
00012 public:
00013     static std::unique_ptr<Angajat> creeazaAngajat(const std::string& tip, const std::string& nume,
00014         const std::string& prenume, const std::string& cnp, const Data& dataAngajarii, const std::string&
00015         orasDomiciliu, const std::map<std::string, std::set<std::string>>& specializari = {});
00016     static std::unique_ptr<Angajat> creeazaDinCSV(const std::string& linie);
00017
00018 private:
00019     static std::vector<std::string> splitLine(const std::string& linie, char delimitator);
00020 };

```

5.3 AngajatiManager.h

```

00001 #pragma once
00002 #include <vector>
00003 #include <memory>
00004 #include <string>
00005 #include "Angajat.h"
00006 #include "Tehnician.h"
00007
00008 class AngajatiManager {
00009 private:
00010     std::vector<std::unique_ptr<Angajat>> angajati;
00011
00012 public:
00013     AngajatiManager() = default;
00014
00015     void incarcaDinCSV(const std::string& numeFisier);
00016
00017     Tehnician* gasesteTehnicianDisponibil(const std::string& tipAparat, const std::string&
00018         marcaAparat);
00019
00020
00021     void adaugaAngajat(std::unique_ptr<Angajat> a);
00022     bool stergeAngajat(const std::string& cnp);
00023     bool modificaNume(const std::string& cnp, const std::string& numeNou, const std::string&
00024         prenumeNou);
00025     Angajat* cautaDupaCNP(const std::string& cnp);
00026     Angajat* cautaDupaID(int id);
00027
00028
00029
00030
00031     bool verificaPersonalMinim() const; // 3 Tech, 1 Rec, 1 Sup
00032
00033     bool adaugaAngajatManual(std::string tip, std::string nume, std::string prenume, std::string cnp,
00034         std::string data, std::string oras);
00035
00036     void afiseazaToti() const;
00037
00038     std::vector<Angajat*> getTop3Salarii() const;
00039
00040     void genereazaRaportTop3Salarii(const std::string& numeFisier) const;
00041     Tehnician* getTehnicianCeaMaiLungaReparatie() const;
00042
00043
00044 };

```

5.4 CerereFactory.h

```

00001 #pragma once
00002 #include <string>
00003 #include <memory>
00004 #include <vector>
00005 #include "CerereReparatie.h"
00006
00007 class CerereFactory {
00008 public:
00009     static std::unique_ptr<CerereReparatie> creeazaDinCSV(const std::string& linie);
00010
00011 private:
00012     static std::vector<std::string> splitLine(const std::string& linie, char delimitator);
00013 };
00014

```

5.5 CerereManager.h

```

00001 #pragma once
00002 #include <vector>
00003 #include <queue>
00004 #include <memory>
00005 #include <string>
00006 #include "CerereReparatie.h"
00007 #include "ElectrocasniceManager.h"
00008
00009 class CerereManager {
00010 private:
00011     std::vector<std::unique_ptr<CerereReparatie>> toateCererile;
00012
00013     std::queue<CerereReparatie*> coadaAsteptare;
00014     std::vector<CerereReparatie*> cereriInLucru;
00015     std::vector<CerereReparatie*> cereriFinalizate;
00016
00017 public:
00018     CerereManager() = default;
00019
00020     void incarcaDinCSV(const std::string& fisier, ElectrocasniceManager& catalog);
00021     void adaugaCerere(std::unique_ptr<CerereReparatie> c);
00022
00023     CerereReparatie* getUrmatoareaInAsteptare();
00024     void popAsteptare();
00025     void puneInapoiInAsteptare(CerereReparatie* c);
00026     bool areCereriInAsteptare() const;
00027
00028     void repartizeazaCerere(CerereReparatie* c, int idTehnician);
00029     void mutaInFinalizate(CerereReparatie* c);
00030
00031     std::vector<CerereReparatie*>& getCereriInLucru();
00032     bool areCereriActive() const;
00033     void afiseazaStatisticiCurente() const;
00034
00035
00036
00037     std::vector<int> getIdsInAsteptare() const;
00038
00039     void genereazaRaportCereriInAsteptare(const std::string& numeFisier) const;
00040
00041     void afiseazaIstoricReparatii() const;
00042
00043
00044
00045 };

```

5.6 CerereReparatie.h

```

00001 #pragma once
00002 #include "Electrocasnici.h"
00003 #include "Tehnician.h"
00004 #include <string>
00005 #include <iostream>
00006 #include <memory>
00007 #include <ctime>
00008
00009 enum class StareCerere{
0010     IN_ASTEPTARE,
0011     REPARTIZATA,
0012

```

```

00013     IN_LUCRU,
00014     FINALIZATA,
00015     INVALIDA
00016 };
00017
00018
00019 class CerereReparatie {
00020     //TODO la fel! alta denumire pentru id uri!!
00021     static int nextID;
00022     int id;
00023     std::unique_ptr<Electrocasnic> aparat;
00024     time_t timestamp;
00025     int nivelComplexitate; // 0-5 (0- nu se poate repara! REMAT!!)
00026     int durataEstimata;
00027     int durataRamasa;
00028     double pretReparatie;
00029     StareCerere stare;
00030     int idTehnician; // -1 daca e nerepartizata
00031
00032 public:
00033     CerereReparatie()=default;
00034     CerereReparatie(std::unique_ptr<Electrocasnic> _aparat, const time_t _timestamp, int
00035     _nivelComplexitate);
00036
00037     // constructor de copiere si operatorul egal pentru unique_ptr ???????
00038
00039     CerereReparatie(const CerereReparatie&);
00040     CerereReparatie& operator=(const CerereReparatie& );
00041
00042     //getteri
00043     int getId() const;
00044     int getNivelComplexitate() const;
00045     const Electrocasnic* getAparat() const;
00046     time_t getTimeStamp() const;
00047     int getDurataEstimata() const;
00048     int getDurataRamasa() const;
00049     double getPretReparatie() const;
00050     StareCerere getStare() const;
00051     int getIdTehnician() const;
00052
00053
00054     // setters
00055
00056     void setStare(const StareCerere);
00057     void setIdTehnician(int);
00058
00059
00060     // Procesare
00061
00062     void proceseaza(); // reduce durata ramasa
00063     bool esteFinalizata() const;
00064     bool esteValida() const;
00065
00066     void afisare(std::ostream&) const;
00067     std::string getTimeStampString() const;
00068
00069
00070
00071     //Comparare pentru sortare???
00072     bool operator<(const CerereReparatie& ) const;
00073
00074 };
00075

```

5.7 CNPValidator.h

```

00001 #pragma once
00002 #include "Data.h"
00003 #include <string>
00004
00005 class CNPValidator {
00006 public:
00007     static bool esteValid(const std::string& cnp);
00008     static Data getDataNasterii(const std::string& cnp);
00009     static char getSex(const std::string& cnp);
00010
00011 private:
00012
00013     static bool verificaCifreControl(const std::string& cnp);
00014     static bool verificaData(int an, int luna, int zi);
00015
00016     CNPValidator() = delete;
00017 };

```

5.8 Data.h

```

00001 #pragma once
00002 #include <iostream>
00003 #include <string>
00004 #include <ostream>
00005 #include <iostream>
00006
00007 class Data {
00008     int zi, luna, an;
00009
0010 public:
0011     Data()=default;
0012     Data(int, int, int);
0013     Data(const std::string&); // "DD.MM.YYYY"
0014
0015     // VALIDARI
0016     bool esteValida() const;
0017
0018
0019     //getteri
0020     int getVarsta() const;
0021     int getZi() const;
0022     int getLuna() const;
0023     int getAn() const;
0024
0025     //operatori
0026     bool operator<(const Data&) const;
0027     bool operator>(const Data&) const;
0028     bool operator==(const Data&) const;
0029     friend std::ostream& operator<<(std::ostream&, const Data&);
0030
0031     //static
0032     static Data dataCurenta();
0033
0034     // string toString() const
0035
0036
0037
0038 };

```

5.9 Electrocasic.h

```

00001 #pragma once
00002 #include <string>
00003 #include <memory>
00004 #include "Data.h"
00005
00006 class Electrocasic {
00007 protected:
00008     std::string tip, marca, model;
00009     int anFabricatie;
00010     double pretCatalog;
00011
00012 public:
00013     Electrocasic() = default;
00014     Electrocasic(const std::string&, const std::string&, const std::string&, int, double);
00015
00016     virtual ~Eelectrocasic() = default;
00017
00018     // getteri
00019     std::string getTip() const;
00020     std::string getMarca() const;
00021     std::string getModel() const;
00022     int getAnFabricatie() const;
00023     double getPretCatalog() const;
00024
00025     int getVechime() const;
00026
00027     virtual void afisare(std::ostream&) const = 0;
00028     virtual std::unique_ptr<Eelectrocasic> cloneaza() const = 0;
00029
00030     bool operator==(const Electrocasic& altul) const;
00031
00032 };

```

5.10 ElectrocasicManager.h

```
00001 #pragma once
```

```

00002 #include <map>
00003 #include <string>
00004 #include <memory>
00005 #include <vector>
00006 #include "Electrocasnic.h"
00007
00008 class ElectrocasniceManager {
00009 private:
0010     std::map<std::string, std::map<std::string, std::map<std::string, std::unique_ptr<Electrocasnic>>>
0011         catalog;
0012     std::map<std::string, int> aparateNereparabile;
0013
0014 public:
0015     ElectrocasniceManager() = default;
0016
0017     void incarcaDinCSV(const std::string& numeFisier);
0018
0019     void adaugaInCatalog(std::unique_ptr<Electrocasnic> e);
0020
0021     bool existaModel(const std::string& tip, const std::string& marca, const std::string& model)
0022         const;
0023     const Electrocasnic* getDetaliiModel(const std::string& tip, const std::string& marca, const
0024         std::string& model) const;
0025
0026
0027     void adaugaModelManual(std::string tip, std::string marca, std::string model, int an, double
0028         pret);
0029     bool stergeModel(std::string marca, std::string model);
0030
0031     void inregistreazaCerereInvalida(const std::string& tip, const std::string& marca, const
0032         std::string& model);
0033     void afiseazaAparateNereparabile() const;
0034     void afiseazaCatalog() const;
0035 };

```

5.11 ElectrocasnicFactory.h

```

00001 #pragma once
00002 #include <string>
00003 #include <memory>
00004 #include <vector>
00005 #include "Electrocasnic.h"
00006
00007 class ElectrocasnicFactory {
00008 public:
00009     static std::unique_ptr<Electrocasnic> creeazaDinCSV(const std::string& linie);
0010
0011 private:
0012     static std::vector<std::string> splitLine(const std::string& linie, char delimitator);
0013 };

```

5.12 Frigider.h

```

00001 #pragma once
00002 #include "Electrocasnic.h"
00003 #include <string>
00004 #include <memory>
00005
00006
00007 class Frigider: public Electrocasnic{
00008
00009     bool areCongelator;
0010
0011 public:
0012     Frigider()=default;
0013     Frigider(const std::string&, const std::string&, int, double, bool);
0014     bool getAreCongelator() const;
0015     void afisare(std::ostream &) const override;
0016     std::unique_ptr<Electrocasnic> cloneaza() const override;
0017
0018 };

```

5.13 MasinaDeSpalat.h

```
00001 #pragma once
00002 #include "Electrocasnic.h"
00003
00004 #include <string>
00005 #include <memory>
00006
00007 class MasinaDeSpalat: public Electrocasnic {
00008     double capacitate; // in kg
00009
00010
00011 public:
00012     MasinaDeSpalat()=default;
00013     MasinaDeSpalat(const std::string&, const std::string&, int, double, double);
00014     double getCapacitate() const;
00015     void afisare(std::ostream&) const override;
00016     std::unique_ptr<Electrocasnic> cloneaza() const override;
00017
00018 };
```

5.14 Meniu.h

```
00001 #pragma once
00002 #include "ServiceManager.h"
00003
00004 class Meniu {
00005 private:
00006     ServiceManager* service;
00007
00008     void clearScreen();
00009     void pause();
00010
00011     void afiseazaTitlu();
00012     void meniuGestiuneAngajati();
00013     void meniuGestiuneElectrocasnice();
00014     void meniuProcesareCereri();
00015     void meniuRaportari();
00016
00017 public:
00018     Meniu();
00019     void ruleazaAplicatie();
00020 };
```

5.15 Receptioner.h

```
00001 #pragma once
00002
00003 #include "Angajat.h"
00004 #include <string>
00005 #include <vector>
00006
00007 class Receptioner: public Angajat{
00008     std::vector<int> idCereriInregistrate;
00009
00010
00011 public:
00012     Receptioner()=default;
00013     Receptioner(const std::string&, const std::string&, const std::string&, const Data&, const std::string&);
00014     void adaugaCerere(int );
00015     const std::vector<int>& getCereri() const;
00016     double calculeazaSalariu() const override;
00017     void afisare(std::ostream&) const override;
00018     std::string getTipAngajat() const override;
00019
00020 };
```

5.16 ServiceManager.h

```
00001 #pragma once
00002 #include "AngajatiManager.h"
00003 #include "ElectrocasniceManager.h"
00004 #include "CerereManager.h"
00005
```

```

00006 class ServiceManager {
00007     private:
00008     ServiceManager() = default;
00009
0010     AngajatiManager angajatiManager;
0011     ElectrocasniceManager electrocasniceManager;
0012     CerereManager cerereManager;
0013
0014     int timpCurent = 0;
0015
0016 public:
0017     static ServiceManager& getInstance() {
0018         static ServiceManager instance;
0019         return instance;
0020     }
0021
0022     ServiceManager(const ServiceManager&) = delete;
0023     ServiceManager& operator=(const ServiceManager&) = delete;
0024
0025
0026     void incarcaDate(const std::string& fAng, const std::string& fEl, const std::string& fCer);
0027
0028     // Simulare
0029     void runSimulare();
0030
0031     // Getteri
0032     AngajatiManager& getAngajatiManager() { return angajatiManager; }
0033     ElectrocasniceManager& getElectrocasniceManager() { return electrocasniceManager; }
0034     CerereManager& getCerereManager() { return cerereManager; }
0035
0036 private:
0037     void proceseazaTic();
0038     void incearcaAlocareCereri();
0039 };

```

5.17 Supervizor.h

```

00001 #pragma once
00002
00003 #include "Angajat.h"
00004 #include <string>
00005 #include <vector>
00006
00007
00008 class Supervizor: public Angajat{
00009
0010     static constexpr double SPOR_CONDUCERE_PROCENT=20.0;
0011
0012     public:
0013
0014     Supervizor()=default;
0015     Supervizor(const std::string&, const std::string&, const std::string&, const Data&, const
0016     std::string&);
0017     double calculeazaSalariu() const override;
0018     void afisare(std::ostream& ) const override;
0019     std::string getTipAngajat() const override;
0020
0021
0022
0023
0024
0025 };

```

5.18 Tehnician.h

```

00001 #pragma once
00002 #include "Angajat.h"
00003 #include <map>
00004 #include <set>
00005 #include <vector>
00006
00007 class Tehnician : public Angajat {
00008
00009     std::map<std::string, std::set<std::string>> specializari;
0010
0011     std::vector<int> idCereriActive;
0012     double valoareaTotalaReparatii;
0013     double durataTotalaLucrata;

```

```

00014
00015     static constexpr double BONUS_REPARATII_PROCENT = 2.0;
00016     static constexpr int MAX_CERERI_ACTIVE = 3;
00017
00018 public:
00019     Tehnician() = default;
00020     Tehnician(const std::string& _nume, const std::string& _prenume, const std::string& _cnp,
00021     const Data& _dataAngajarii, const std::string& _orasDomiciliu);
00022
00023     // Specializari
00024     void adaugaSpecializare(const std::string& tip, const std::string& marca);
00025     bool areSpecializare(const std::string& tip, const std::string& marca) const;
00026
00027     // Gestiune Cereri Active
00028     bool poatePrimiCerere() const;
00029     void adaugaIdCerereActivă(int id);
00030     void finalizareCerere(int idCerere, double valoareReparatie, int durataEfectiva);
00031
00032     // Getteri pentru alocare automata
00033     double getDurataTotalaLucrata() const;
00034     int getNrCereriActive() const ;
00035
00036     // Salariu si Afisare
00037     double calculeazaSalariu() const override;
00038     void afisare(std::ostream& ) const override;
00039     std::string getTipAngajat() const override;
00040 };

```

5.19 TV.h

```

00001 #pragma once
00002 #include "Electrocasnic.h"
00003
00004 #include <string>
00005 #include <memory>
00006 #include <iostream>
00007
00008 class TV: public Electrocasnic{
00009
0010     double diagonala;
0011     bool esteInCm; // true=cm si false=incii
0012
0013 public:
0014     TV()=default;
0015     TV(const std::string&, const std::string&, int, double, double, bool);
0016
0017     double getDiagonala() const;
0018     bool getEsteInCm() const;
0019     void afisare(std::ostream&) const override;
0020
0021     std::unique_ptr<Electrocasnic> cloneaza() const override;
0022
0023 };

```


Index

afisare
 Frigider, 14
 MasinaDeSpalat, 16
 Receptioner, 18
 Supervizor, 21
 Tehnician, 23
 TV, 25
Angajat, 7
AngajatFactory, 8
AngajatiManager, 9

calculeazaSalariu
 Receptioner, 18
 Supervizor, 21
 Tehnician, 23
CerereFactory, 9
CerereManager, 9
CerereReparatie, 10
cloneaza
 Frigider, 14
 MasinaDeSpalat, 16
 TV, 25
CNPValidator, 10

Data, 11

Electrocasnic, 11
ElectrocasniceManager, 12
ElectrocasnicFactory, 13

Frigider, 13
 afisare, 14
 cloneaza, 14

getTipAngajat
 Receptioner, 18
 Supervizor, 21
 Tehnician, 23

MasinaDeSpalat, 15
 afisare, 16
 cloneaza, 16
Meniu, 16

Receptioner, 17
 afisare, 18
 calculeazaSalariu, 18
 getTipAngajat, 18

ServiceManager, 19
Supervizor, 19