My Project

Generated by Doxygen 1.9.8

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

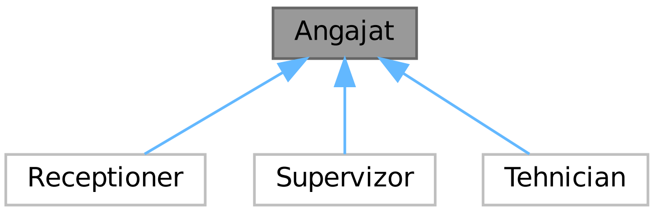
Here is a list of all documented files with brief descriptions:

Chapter 4

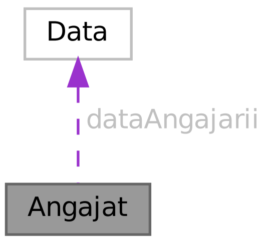
Class Documentation

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 **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
- 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 **puneInapoiInAsteptare** (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 **afiseazaIstoricReparatii** () 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< Electrocasnic > _aparat, const time_t _timestamp, int _nivel↵ Complexitate)
- **CerereReparatie** (const CerereReparatie &)
- CerereReparatie & **operator=** (const CerereReparatie &)
- int **getId** () const
- int **getNivelComplexitate** () const
- const Electrocasnic ∗ **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 **getDataNasterii** (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

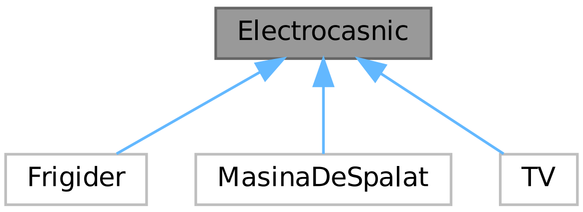
- std::ostream & **operator**<< (std::ostream &, const Data &)

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

- Data.h
- Data.cpp

4.9 Electrocasnic Class Reference

Inheritance diagram for Electrocasnic:



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

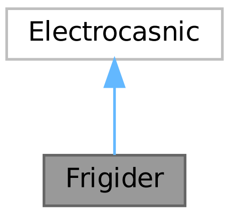
- static std::unique_ptr< Electrocasnic > **creeazaDinCSV** (const std::string &linie)

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

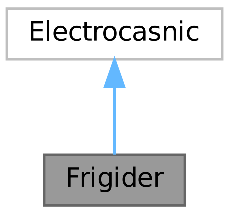
- ElectrocasnicFactory.h
- ElectrocasnicFactory.cpp

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 Electrocasnic

- **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
- bool **operator==** (const Electrocasnic &altul) const

Additional Inherited Members

Protected Attributes inherited from Electrocasnic

- 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 Electrocasnic.

4.12.1.2 cloneaza()

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

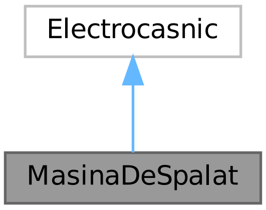
Implements Electrocasnic.

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

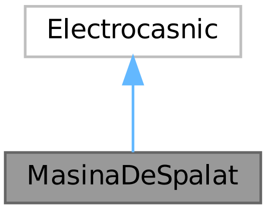
- Frigider.h
- Frigider.cpp

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< Electrocasnic > cloneaza () const override

Public Member Functions inherited from Electrocasnic

- **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
- bool **operator==** (const Electrocasnic &altul) const

Additional Inherited Members

Protected Attributes inherited from Electrocasnic

- 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 Electrocasnic.

4.13.1.2 cloneaza()

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

Implements Electrocasnic.

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

- MasinaDeSpalat.h
- MasinaDeSpalat.cpp

4.14 Meniu Class Reference

Public Member Functions

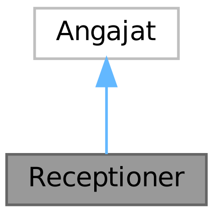
- void **ruleazaAplicatie** ()

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

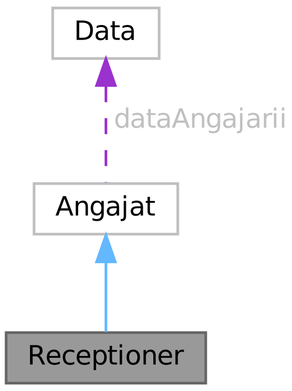
- Meniu.h
- Meniu.cpp

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

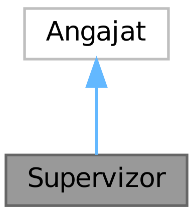
- static ServiceManager & **getInstance** ()

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

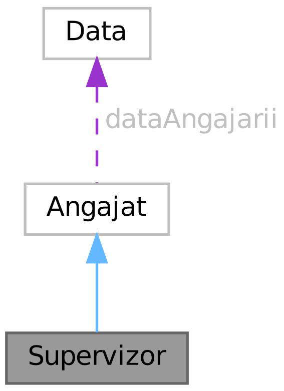
- ServiceManager.h
- ServiceManager.cpp

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]
```

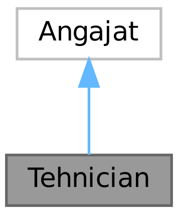
Implements Angajat.

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

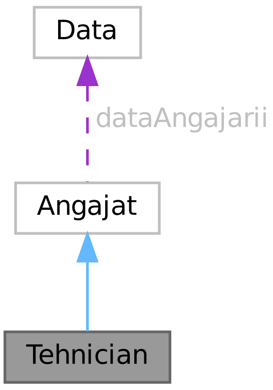
- Supervizor.h
- Supervizor.cpp

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 **adaugaIdCerereActiva** (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 **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

4.18.1 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]
```

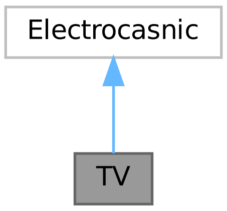
Implements Angajat.

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

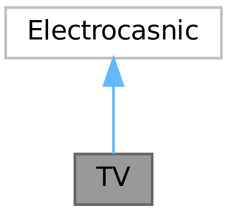
- Tehnician.h
- Tehnician.cpp

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 **getEsteInCm** () const
- void afisare (std::ostream &) const override
- std::unique_ptr< Electrocasnic > cloneaza () const override

Public Member Functions inherited from Electrocasnic

- **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
- bool **operator==** (const Electrocasnic &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

```cpp
#pragma once
#include <string>
#include <iostream>
#include "Data.h"

class Angajat{

protected:

    //TODO : gandeste te la o denumire mai buna pentru id urile de la angajat si cerere
    static int nextID;
    const int id;
    std::string nume, prenume, cnp;
    Data dataAngajarii;
    std::string orasDomiciliu;

    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;

public:

    Angajat():id(0){}
    Angajat(const std::string& ,const std::string&, const std::string& , const Data& , const std::string& );

    virtual ~Angajat()=default;


    // getteri

    int getId() const;
    std::string getNume() const;
    std::string getPrenume() const;
    std::string getCNP() const;
    Data getDataAngajarii() const;
    std::string getOrasDomiciliu() const;


    //setteri

    void setNume(const std::string&);
    void setPrenume(const std::string& );


    //validare

    void validareNume(const std::string& , const std::string& ) const;


    //calcul salariu-- VIRTUAL PURE
    virtual double calculeazaSalariu() const=0;
    virtual void afisare(std::ostream& ) const;

        //tip angajat-- VIRTUAL PURE
    virtual std::string getTipAngajat() const=0;

    // auxiliare
    double calculeazaBonusFidelitate() const;
    bool primestePrimaTransport() const;

};
```

5.2 AngajatFactory.h

```


#pragma once
#include <string>
#include <memory>
#include <vector>
#include <map>
#include <set>
#include "Angajat.h"

class AngajatFactory {
public:
    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);

private:
    static std::vector<std::string> splitLine(const std::string& linie, char delimitator);
};
```

5.3 AngajatiManager.h

```
#pragma once
#include <vector>
#include <memory>
#include <string>
#include "Angajat.h"
#include "Tehnician.h"

class AngajatiManager {
private:
    std::vector<std::unique_ptr<Angajat» angajati;

public:
    AngajatiManager() = default;

    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; // 3 Tech, 1 Rec, 1 Sup

    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;


};
```

5.4 CerereFactory.h

```
#pragma once
#include <string>
#include <memory>
#include <vector>
#include "CerereReparatie.h"

class CerereFactory {
public:
    static std::unique_ptr<CerereReparatie> creeazaDinCSV(const std::string& linie);

private:
    static std::vector<std::string> splitLine(const std::string& linie, char delimitator);
};

```

5.5 CerereManager.h

```
#pragma once
#include <vector>
#include <queue>
#include <memory>
#include <string>
#include "CerereReparatie.h"
#include "ElectrocasniceManager.h"

class CerereManager {
private:
    std::vector<std::unique_ptr<CerereReparatie» toateCererile;

    std::queue<CerereReparatie*> coadaAsteptare;
    std::vector<CerereReparatie*> cereriInLucru;
    std::vector<CerereReparatie*> cereriFinalizate;

public:
    CerereManager() = default;

    void incarcaDinCSV(const std::string& fisier, ElectrocasniceManager& catalog);
    void adaugaCerere(std::unique_ptr<CerereReparatie> c);

    CerereReparatie* getUrmatoareaInAsteptare();
    void popAsteptare();
    void puneInapoiInAsteptare(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 afiseazaIstoricReparatii() const;


};
```

5.6 CerereReparatie.h

```
#pragma once
#include "Electrocasnic.h"
#include "Tehnician.h"
#include <string>
#include <iostream>
#include <memory>
#include <ctime>

enum class StareCerere{

    IN_ASTEPTARE,
    REPARTIZATA,
    IN_LUCRU,
    FINALIZATA,
    INVALIDA
};


    class CerereReparatie {
        //TODO la fel! alta denumire pentru id uri!!
        static int nextID;
         int id;
        std::unique_ptr<Electrocasnic> aparat;
        time_t timestamp;
        int nivelComplexitate;  // 0-5 (0- nu se poatte repara! REMAT!!)
        int durataEstimata;
        int durataRamasa;
        double pretReparatie;
        StareCerere stare;
        int idTehnician; // -1 daca e nerepartizata

        public:
            CerereReparatie()=default;
            CerereReparatie(std::unique_ptr<Electrocasnic> _aparat, const time_t _timestamp, int
_nivelComplexitate);


            // constructor de copiere si operatorul egal pentru unique_ptr ????????

            CerereReparatie(const CerereReparatie&);
            CerereReparatie& operator=(const CerereReparatie& );

            //getteri
            int getId() const;
            int getNivelComplexitate() const;
            const Electrocasnic* getAparat() const;
            time_t getTimeStamp() const;
            int getDurataEstimata() const;
            int getDurataRamasa() const;
            double getPretReparatie() const;
            StareCerere getStare() const;
            int getIdTehnician() const;


            // setters

            void setStare(const StareCerere);
            void setIdTehnician(int);


            // Procesare

            void proceseaza(); // reduce durata ramasa
            bool esteFinalizata() const;
            bool esteValida() const;

        void afisare(std::ostream&) const;
        std::string getTimeStampString() const;



        //Comparare pentru sortare????
        bool operator<(const CerereReparatie& ) const;

    };

```

5.7 CNPValidator.h

```
#pragma once
#include "Data.h"
#include <string>

class CNPValidator {
public:
    static bool esteValid(const std::string& cnp);
    static Data getDataNasterii(const std::string& cnp);
    static char getSex(const std::string& cnp);

private:

    static bool verificaCifreControl(const std::string& cnp);
    static bool verificaData(int an, int luna, int zi);

    CNPValidator() = delete;
};
```

5.8 Data.h

```cpp
#pragma once
#include <iostream>
#include <string>
#include <ostream>
#include <istream>

class Data {
    int zi, luna, an;

public:
    Data()=default;
    Data(int, int, int);
    Data(const std::string& ); // "DD.MM.YYYY"

    // VALIDARI
    bool esteValida() const;


    //getteri
    int getVarsta() const;
    int getZi() const;
    int getLuna() const;
    int getAn() const;

    //operatori
    bool operator<(const Data&) const;
    bool operator>(const Data&) const;
    bool operator==(const Data&) const;
    friend std::ostream& operator«(std::ostream&, const Data&);

    //static
    static Data dataCurenta();

    // string toString() const



};
```

5.9 Electrocasnic.h

```cpp
#pragma once
#include <string>
#include <memory>
#include "Data.h"

class Electrocasnic {
protected:
    std::string tip, marca, model;
    int anFabricatie;
    double pretCatalog;

public:
    Electrocasnic() = default;
    Electrocasnic(const std::string& , const std::string& , const std::string& , int , double );

    virtual ~Electrocasnic() = default;

    // getteri
    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;

};
```

5.10 ElectrocasniceManager.h

```cpp
#pragma once
```

```cpp
#include <map>
#include <string>
#include <memory>
#include <vector>
#include "Electrocasnic.h"

class ElectrocasniceManager {
private:
    std::map<std::string, std::map<std::string, std::map<std::string, std::unique_ptr<Electrocasnic»» catalog;

    std::map<std::string, int> aparateNereparabile;

public:
    ElectrocasniceManager() = default;

    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;
};
```

5.11 ElectrocasnicFactory.h

```cpp
#pragma once
#include <string>
#include <memory>
#include <vector>
#include "Electrocasnic.h"

class ElectrocasnicFactory {
public:
    static std::unique_ptr<Electrocasnic> creeazaDinCSV(const std::string& linie);

private:
    static std::vector<std::string> splitLine(const std::string& linie, char delimitator);
};
```

5.12 Frigider.h

```cpp
#pragma once
#include "Electrocasnic.h"
#include <string>
#include <memory>


class Frigider: public Electrocasnic{

    bool areCongelator;

public:
    Frigider()=default;
    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;

};
```

5.13 MasinaDeSpalat.h

```
#pragma once
#include "Electrocasnic.h"

#include  <string>
#include <memory>

class MasinaDeSpalat: public Electrocasnic {

    double capacitate; // in kg

public:
    MasinaDeSpalat()=default;
    MasinaDeSpalat(const std::string& , const std::string&, int,double,double );
    double getCapacitate() const;
    void afisare(std::ostream&) const override;
    std::unique_ptr<Electrocasnic> cloneaza() const override;

};
```

5.14 Meniu.h

```
#pragma once
#include "ServiceManager.h"

class Meniu {
private:
    ServiceManager* service;

    void clearScreen();
    void pause();

    void afiseazaTitlu();
    void meniuGestiuneAngajati();
    void meniuGestiuneElectrocasnice();
    void meniuProcesareCereri();
    void meniuRaportari();

public:
    Meniu();
    void ruleazaAplicatie();
};
```

5.15 Receptioner.h

```
#pragma once

#include "Angajat.h"
#include <string>
#include <vector>

class Receptioner: public Angajat{

    std::vector<int> idCereriInregistrate;

    public:
        Receptioner()=default;
        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;

};
```

5.16 ServiceManager.h

```
#pragma once
#include "AngajatiManager.h"
#include "ElectrocasniceManager.h"
#include "CerereManager.h"

```

```
class ServiceManager {
private:
    ServiceManager() = default;

    AngajatiManager angajatiManager;
    ElectrocasniceManager electrocasniceManager;
    CerereManager cerereManager;

    int timpCurent = 0;

public:
    static ServiceManager& getInstance() {
        static ServiceManager instance;
        return instance;
    }

    ServiceManager(const ServiceManager&) = delete;
    ServiceManager& operator=(const ServiceManager&) = delete;


    void incarcaDate(const std::string& fAng, const std::string& fEl, const std::string& fCer);

    // Simulare
    void runSimulare();

    // Getteri
    AngajatiManager& getAngajatiManager() { return angajatiManager; }
    ElectrocasniceManager& getElectrocasniceManager() { return electrocasniceManager; }
    CerereManager& getCerereManager() { return cerereManager; }

private:
    void proceseazaTic();
    void incearcaAlocareCereri();
};
```

5.17 Supervizor.h

```
#pragma once

#include "Angajat.h"
#include <string>
#include <vector>


class Supervizor: public Angajat{

    static constexpr double SPOR_CONDUCERE_PROCENT=20.0;

    public:

        Supervizor()=default;
        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;






};
```

5.18 Tehnician.h

```
#pragma once
#include "Angajat.h"
#include <map>
#include <set>
#include <vector>

class Tehnician : public Angajat {

    std::map<std::string, std::set<std::string» specializari;

    std::vector<int> idCereriActive;
    double valoareaTotalaReparatii;
    double durataTotalaLucrata;

    static constexpr double BONUS_REPARATII_PROCENT = 2.0;
    static constexpr int MAX_CERERI_ACTIVE = 3;

public:
        Tehnician() = default;
        Tehnician(const std::string& _nume, const std::string& _prenume, const std::string& _cnp,
const Data& _dataAngajarii, const std::string& _orasDomiciliu);

        // Specializari
        void adaugaSpecializare(const std::string& tip, const std::string& marca);
        bool areSpecializare(const std::string& tip, const std::string& marca) const;

        // Gestiune Cereri Active
        bool poatePrimiCerere() const;
        void adaugaIdCerereActiva(int id);
        void finalizareCerere(int idCerere, double valoareReparatie, int durataEfectiva);

        // Getteri pentru alocare automata
        double getDurataTotalaLucrata() const;
        int getNrCereriActive() const ;

        // Salariu si Afisare
        double calculeazaSalariu() const override;
        void afisare(std::ostream& ) const override;
        std::string getTipAngajat() const override;
};
```

5.19 TV.h

```
#pragma once
#include "Electrocasnic.h"

#include <string>
#include <memory>
#include <iostream>

class TV: public Electrocasnic{

    double diagonala;
    bool esteInCm; // true=cm si false=inciii

public:
    TV()=default;
    TV(const std::string&, const std::string&,int,double,double,bool);

    double getDiagonala() const;
    bool getEsteInCm() const;
    void afisare(std::ostream&) const override;

    std::unique_ptr<Electrocasnic> cloneaza() const override;

};
```

Index