PARPA_Z4_PZSP2

Generated by Doxygen 1.9.5

Chapter 1

Projekt

Projekt: System ADMS, będący w stanie na bieżąco reagować na sygnały otrzymywane od operatora sieci elektrycznej umożliwiający ładowanie samochodów elektrycznych na parkingu.

Struktura projektu:

- algorithm - folder dla programu z algorytmem balansującym energię elektryczną

  Uruchomienie:

- database - folder ze schematami bazy danych
- application - folder z backendem oraz frontendem aplikacji

  Uruchomienie:

1. Z głównego folderu repozytorium należy przejść do folderu application\frontend:

   cd .\application\frontend\

2. Przy pierwszym uruchomieniu po pobraniu należy zacząć od pobrania zależności:

   npm install

3. Uruchomienie Reacta:

   npm start

4. Uruchomienie Flaska (w drugim terminalu, również w folderze application\frontend):

   npm run start-api

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

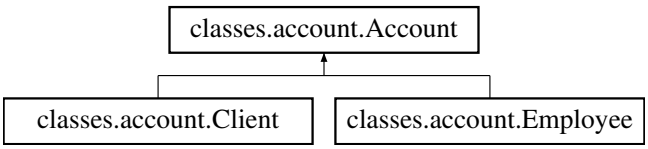
Here is a list of all documented files with brief descriptions:

Chapter 5

Class Documentation

5.1 classes.account.Account Class Reference

Inheritance diagram for classes.account.Account:



Public Member Functions

- def __init__ (self, str username, str password, str mail, str phone_no)
- bool check_password (self, str pwd)
- int get_id (self)

Static Public Member Functions

- str get_type (str username)

Public Attributes

- **username**
- **mail**
- **phone_no**

5.1.1 Detailed Description

```
Class representing a basic account
```

5.1.2 Constructor & Destructor Documentation

5.1.2.1 __init__()

```
def classes.account.Account.__init__ (
              self,
             str username,
             str password,
             str mail,
             str phone_no )
```

```
Args:
    username:   Unique username for the client account.
    password:   Account's password.
    mail:       Mail associated with the account.
    phone_no:   Phone number to the client account.
```

Reimplemented in classes.account.Client, and classes.account.Employee.

5.1.3 Member Function Documentation

5.1.3.1 check_password()

```
bool classes.account.Account.check_password (
              self,
             str pwd )
```

```
Checking if account can be authenticated.
If password provided by the user matches the one in the database.

Args:
    pwd:        Provided password to be verified.

Returns:
    True:       When password is correct.
    False:      When password is incorrect.
```

5.1.3.2 get_id()

```
int classes.account.Account.get_id (
              self )
```

```
Get account unique id.

Returns:
    Account's primary key from database.
```

5.1.3.3 get_type()

```
str classes.account.Account.get_type (
            str username )  [static]
```

```
Get account's type by username.

Returns:
    Account's type: EMPLOYEE or CLIENT.
```

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

- application/classes/account.py

5.2 balancer.Balancer Class Reference

Public Member Functions

- def balance (self, Carpark carpark, energy_usage_demand)

Public Attributes

- **usage**

Static Public Attributes

- int **usage** = 0

5.2.1 Member Function Documentation

5.2.1.1 balance()

```
def balancer.Balancer.balance (
             self,
            Carpark carpark,
             energy_usage_demand )
```

```
Sets chargers' power to satisfy energy usage demand.

Args:
    carpark:                Carpark to preform balancing on..
    Energy_usage_demand:    energy usage requested by operator.

Returns:
    Approximate energy to be used by given carpark.
```

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

- algorithm/classes/balancer.py

5.3 classes.car.Car Class Reference

Public Member Functions

- def __init__ (self, str vin, str reg_no, str model, str brand, float capacity, int owner_id)
- bool is_parked (self)
- None park (self, float charge_level, str charger_id, datetime departure_time=None)
- None change_departure (self, datetime new_time)
- None unpark (self)

Static Public Member Functions

- Car add_car (account.Client client, str vin, str reg_no, str model, str brand, float capacity)
- Car get_car (str vin)
- list[Car] get_client_cars (account.Client client)

Public Attributes

- **vin**
- **reg_no**
- **model**
- **brand**
- **capacity**
- **owner_id**

5.3.1 Detailed Description

```
Class representing a single car
```

5.3.2 Constructor & Destructor Documentation

5.3.2.1 __init__()

```
def classes.car.Car.__init__ (
              self,
             str vin,
             str reg_no,
             str model,
             str brand,
             float capacity,
             int owner_id )
```

```
Args:
    vin:        VIN number of the car.
    reg_no:     Registration number of the car.
    model:      Model of the car.
    brand:      Brand of the car.
    capacity:   Maximum capacity of car's tank.
    owner_id:   ID of the owner client.
```

5.3.3 Member Function Documentation

5.3.3.1 add_car()

```
Car classes.car.Car.add_car (
            account.Client client,
            str vin,
            str reg_no,
            str model,
            str brand,
            float capacity )  [static]
```

```
Adds a new car to the system.
Car has to be already owned by a client.

Args:
    client:     Client owner of the car.
    vin:        VIN number of the car.
    reg_no:     Registration number of the car.
    model:      Model of the car.
    brand:      Brand of the car.
    capacity:   Maximum capacity of car's tank.

Returns:
    A new Car object.
```

5.3.3.2 change_departure()

```
None classes.car.Car.change_departure (
             self,
            datetime new_time )
```

```
Changes planned departure time of a parked car.

Args:
    new_time:         New planned departure time.

Raises:
    Exception:        When chosen car is not parked.
```

5.3.3.3 get_car()

```
Car classes.car.Car.get_car (
            str vin )  [static]
```

```
Fetches a car with given VIN number.

Args:
    vin:        VIN number of searched car.

Returns:
    A new car object.
```

5.3.3.4 get_client_cars()

```
list[Car] classes.car.Car.get_client_cars (
            account.Client client )  [static]
```

Fetches all cars belonging to given client's account.

```
Args:
    client:     Client's account to search for cars.

Returns:
    A list of cars belonging to the client.
```

5.3.3.5 is_parked()

```
bool classes.car.Car.is_parked (
             self )
```

Checks if given car is parked or not.

```
Returns:
    True:   If car is parked.
    False:  If car is not parked.
```

5.3.3.6 park()

```
None classes.car.Car.park (
             self,
            float charge_level,
            str charger_id,
            datetime  departure_time = None )
```

Parks a car updating information in the database.

```
Args:
    charge_level:     Energy level of the parking car.
    charger_id:       Id of the used charger.
    departure_time:   Estimated departure time.

Raises:
    Exception:        When chosen car is already parked.
```

5.3.3.7 unpark()

```
None classes.car.Car.unpark (
              self )


Unparks a car updating information in the database.

Raises:
    Exception:        When chosen car is not parked.
```

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

- application/classes/car.py

5.4 carpark.Carpark Class Reference

Public Member Functions

- def __init__ (self, id)
- str **__str__** (self)
- def **charge** (self)
- def actualize (self, time_period=Decimal(0.25))
- def grade_cars (self)
- def sort_cars (self)
- def calculate_max_energy_use (self)
- def set_all_stations (self, order)
- def calculate_real_energy_use (self)

Public Attributes

- **id**
- **active_charging_stations**
- **current_time**
- **max_energy_usage**

5.4.1 Constructor & Destructor Documentation

5.4.1.1 __init__()

```
def carpark.Carpark.__init__ (
              self,
              id )


Args:
    id:     Parking id.
```

5.4.2 Member Function Documentation

5.4.2.1 actualize()

```
def carpark.Carpark.actualize (
              self,
              time_period = Decimal(0.25) )
```

Actualizes time and active charging stations.

5.4.2.2 calculate_max_energy_use()

```
def carpark.Carpark.calculate_max_energy_use (
              self )
```

Calculated max possible energy usage for whole carpark.

5.4.2.3 calculate_real_energy_use()

```
def carpark.Carpark.calculate_real_energy_use (
              self )
```

Calculates real energy usage of a whole carpark.

Returns:
    Real energy usage of a whole carpark.

5.4.2.4 grade_cars()

```
def carpark.Carpark.grade_cars (
              self )
```

Decides the priority of charging for each parked car.

5.4.2.5 set_all_stations()

```
def carpark.Carpark.set_all_stations (
              self,
              order )
```

```
Sets all stations orders to given value (respecting constraints).

Returns:
    Energy usage of a whole carpark, after seting orders.
```

5.4.2.6 sort_cars()

```
def carpark.Carpark.sort_cars (
              self )
```

```
Sorts cars in active_charging_stations according to their status.
```

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

- algorithm/classes/carpark.py

5.5 char_stat_from_db_creator.Char_stat_from_db_creator Class Reference

Static Public Member Functions

- def **get_charging_stations** (parking_id, time_period)
- def **get_curr_time** ()
- def **insert_new_charging** (Charging_station station)

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

- algorithm/database/char_stat_from_db_creator.py

5.6 charging_car.Charging_car Class Reference

Public Member Functions

- def __init__ (self, charge_level, start_charge_level, pickup_time, car_capacity, charger_type, charger_power, car_vin="")
- str **__str__** (self)

Public Attributes

- **charge_level**
- **start_charge_level**
- **pickup_time**
- **car_capacity**
- **charger_type**
- **charger_power**
- **car_vin**

5.6.1 Constructor & Destructor Documentation

5.6.1.1 __init__()

```
def charging_car.Charging_car.__init__ (
              self,
              charge_level,
              start_charge_level,
              pickup_time,
              car_capacity,
              charger_type,
              charger_power,
              car_vin = "" )
```

```
Args:
    charge_level:           Current charging level of car battery.
    start_charge_level:     Charging level at which car was parked.
    pickup_time:            Time at which car is going to get picked up.
    car_capacity:           Car battery capacity.
    charger_type:           AC or DC.
    charger_power:          Max power of the charger.
```

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

- algorithm/classes/charging_car.py

5.7 charging_station.Charging_station Class Reference

Public Member Functions

- def __init__ (self, Charging_car car, charger_id, carpark_id, time_period=Decimal(0.25))
- str **__str__** (self)
- def set_tags (self, current_time)
- def **set_hour_to_go** (self, current_time)
- def **set_below_start** (self)
- def **charge** (self)
- def set_status (self)
- def energy_usage (self)
- def max_energy_usage (self)
- def set_order (self, order)

Public Attributes

- **car**
- **charger_id**
- **carpark_id**
- **order**
- **below_start**
- **hour_to_go**
- **status**
- **time_period**
- **new_charge_level**

5.7.1 Constructor & Destructor Documentation

5.7.1.1 __init__()

```
def charging_station.Charging_station.__init__ (
              self,
             Charging_car car,
              charger_id,
              carpark_id,
              time_period = Decimal(0.25) )
```

```
Args:
    car:            Charging_car object representing car connected to this station.
    charger_id:     Charging station id.
    carpark_id:     Carpark id.
    time_period:    how long is order going to be set up.
```

5.7.2 Member Function Documentation

5.7.2.1 energy_usage()

```
def charging_station.Charging_station.energy_usage (
              self )
```

```
Calculates charging station's energy usage.

Returns:
    Energy usage.
```

5.7.2.2 max_energy_usage()

```
def charging_station.Charging_station.max_energy_usage (
              self )
```

Calculates max posible charging station's energy usage.

Returns:
    Max posiible energy usage.

5.7.2.3 set_order()

```
def charging_station.Charging_station.set_order (
              self,
              order )
```

Sets station's order to given power, respecting constraints.
Args:
    order: Power on with charging power should now operate.

5.7.2.4 set_status()

```
def charging_station.Charging_station.set_status (
              self )
```

Sets charging station priority to be receive higher order.

5.7.2.5 set_tags()

```
def charging_station.Charging_station.set_tags (
              self,
              current_time )
```

Checks if discharging car would bring it's energy level below starting energy level and sets tag.
Checks if car is about to get picked up in less than an hour and sets appropriate tag.

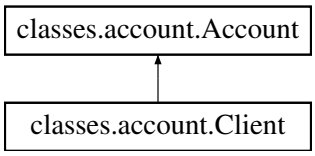
Args:
    current_time: Current time.

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

- algorithm/classes/charging_station.py

5.8 classes.account.Client Class Reference

Inheritance diagram for classes.account.Client:



Public Member Functions

- def __init__ (self, str username, str password, str mail, str phone_no, list cars=0)
- car.Car add_car (self, str vin, str reg_no, str model, str brand, float capacity)
- None save_cars (self, list[car.Car] cars)
- None park_car (self, str vin, float charge_level, int charger_id, datetime departure_time=None)
- None unpark_car (self, str vin)
- None change_car_departure (self, str vin, datetime new_time)

Static Public Member Functions

- Client get_client (str username)
- Client add_client (str username, str password, str mail, str phone_no)

Public Attributes

- **cars**

5.8.1 Detailed Description

```
Class representing a client's account
```

5.8.2 Constructor & Destructor Documentation

5.8.2.1 __init__()

```
def classes.account.Client.__init__ (
              self,
             str username,
             str password,
             str mail,
             str phone_no,
             list  cars = 0 )

Args:
    username, password, mail, phone_no: as in Account
    cars: list of cars added to the client's account
```

Reimplemented from classes.account.Account.

5.8.3 Member Function Documentation

5.8.3.1 add_car()

```
car.Car classes.account.Client.add_car (
             self,
            str vin,
            str reg_no,
            str model,
            str brand,
            float  capacity )
```

Adds provided car and saves it to client.

```
Args:
    vin:        Car's VIN number.
    reg_no:     Registration number of the car.
    model:      Model of the car.
    brand:      Brand of the car.
    capacity:   Maximum capacity of car's tank.

Returns:
    A new Car object.
```

5.8.3.2 add_client()

```
Client classes.account.Client.add_client (
            str username,
            str password,
            str mail,
            str phone_no )  [static]
```

Adds a new client to the database

```
Args:
    username:   Unique username for the client account.
    password:   Account's password.
    mail:       Mail associated with the account.
    phone_no:   Phone number to the client account.

Returns:
    A new Client instance.

Raises:
    UniqueViolation.
```

5.8.3.3 change_car_departure()

```
None classes.account.Client.change_car_departure (
              self,
             str vin,
             datetime new_time )
```

Client can change the planned departure time of parked car.

```
Args:
    vin:        Parked car's VIN number.
```

5.8.3.4 get_client()

```
Client classes.account.Client.get_client (
             str username )  [static]
```

Fetching client information from database using unique username.

```
Args:
    username:        Client's username.

Returns:
    A new Client object.
```

5.8.3.5 park_car()

```
None classes.account.Client.park_car (
              self,
             str vin,
             float charge_level,
             int charger_id,
             datetime  departure_time = None )
```

Enables client to park a given car.

```
Args:
    vin:              Car's VIN number to be parked.
    charge_level:     Energy level of the parking car.
    charger_id:       Id of the used charger.
    departure_time:   Estimated departure time.

Returns:
    None
```

5.8.3.6 save_cars()

```
None classes.account.Client.save_cars (
              self,
             list[car.Car] cars )
```

```
Saves provided list of cars to client's instance.
Only new cars are being added (repetitions are omitted).

Args:
    cars:       List of cars to ba added to client's account.

Returns:
    None.
```

5.8.3.7 unpark_car()

```
None classes.account.Client.unpark_car (
              self,
             str vin )
```

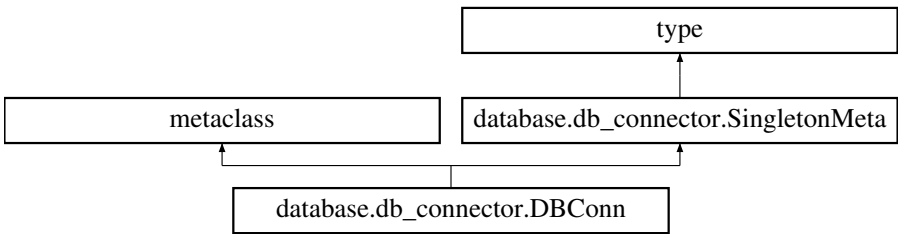
```
Enables client to unpark a given car.

Args:
    vin:        Car's VIN number to be parked.

Returns:
    None
```

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

- application/classes/account.py

5.9 database.db_connector.DBConn Class Reference

Inheritance diagram for database.db_connector.DBConn:



Public Member Functions

- def **__init__** (self)
- def **get_cur** (self)
- def **execute_file** (self, path)
- def **exec_change** (self, stmt)

Public Attributes

- **db_connection**
- **db_cur**

Static Public Attributes

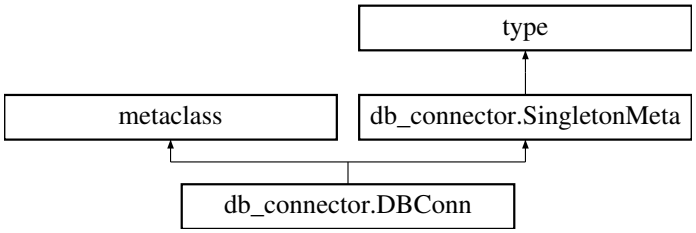
- **str**
- string **user** = "pzsp2"
- string **password** = "parking"
- string **database** = "parpa"

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

- application/database/db_connector.py

5.10 db_connector.DBConn Class Reference

Inheritance diagram for db_connector.DBConn:



Public Member Functions

- def **__init__** (self)
- def **get_cur** (self)
- def **execute_file** (self, path)
- def **exec** (self, stmt)

Public Attributes

- **db_connection**
- **db_cur**

Static Public Attributes

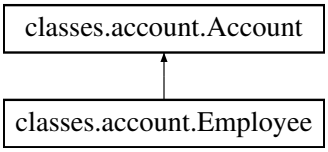
- **str**
- string **user** = "pzsp2"
- string **password** = "parking"
- string **database** = "parpa"

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

- algorithm/database/db_connector.py

5.11 classes.account.Employee Class Reference

Inheritance diagram for classes.account.Employee:



Public Member Functions

- None __init__ (self, str username, str password, str mail, str phone_no, str parking)

Static Public Member Functions

- Employee get_employee (str username)

Public Attributes

- **parking**

5.11.1 Detailed Description

```
Class representing an employee's account.
```

5.11.2 Constructor & Destructor Documentation

5.11.2.1 __init__()

```
None classes.account.Employee.__init__ (
             self,
            str username,
            str password,
            str mail,
            str phone_no,
            str  parking )


Args:
    username, password, mail, phone_no: as in Account
    parking:        Parking where employee works
```

Reimplemented from classes.account.Account.

5.11.3 Member Function Documentation

5.11.3.1 get_employee()

```
Employee classes.account.Employee.get_employee (
            str username )  [static]
```

```
Fetching employee information from database using unique username.

Args:
    username:        Employee's username.

Returns:
    A new Employee object.
```

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

- application/classes/account.py

5.12 operator_mockup.Operator_mockup Class Reference

Public Member Functions

- None **__init__** (self, min, max)
- def **createDemand** (self)

Public Attributes

- **min**
- **max**

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

- algorithm/classes/operator_mockup.py

5.13 classes.parking.Parking Class Reference

Public Member Functions

- def __init__ (self, int places, str city, str street, str addr_nr, str id=0)
- list[car.Car] get_all_cars (self)

Static Public Member Functions

- Parking get_parking (str id)
- Parking get_employee_parking (str username)
- list[Parking] get_all_parkings ()
- list[list] get_parking_map (id)

Public Attributes

- **places**
- **city**
- **street**
- **addr_nr**
- **id**

5.13.1 Detailed Description

```
Class representing a single parking
```

5.13.2 Constructor & Destructor Documentation

5.13.2.1 __init__()

```
def classes.parking.Parking.__init__ (
              self,
             int places,
             str city,
             str street,
             str addr_nr,
             str  id = 0 )
```

```
Args:
    places:     Number of places on the parking.
    city:       City where parking is located.
    street:     Street where parking is located.
    addr_nr:    Address number where parking is located.
    id:         Unique id of the parking.
```

5.13.3 Member Function Documentation

5.13.3.1 get_all_cars()

```
list[car.Car] classes.parking.Parking.get_all_cars (
              self )
```

Getting all parked cars on given parking from db.

Returns:
    List of parked Car objects.

5.13.3.2 get_all_parkings()

```
list[Parking] classes.parking.Parking.get_all_parkings ( )  [static]
```

Getting all available parkings from db.

Returns:
    List of Parking objects.

5.13.3.3 get_employee_parking()

```
Parking classes.parking.Parking.get_employee_parking (
             str username )  [static]
```

Fetching parking using employee's username

Args:
    username: Employee's username working on the parking.

Returns:
    A new Parking object.

5.13.3.4 get_parking()

```
Parking classes.parking.Parking.get_parking (
             str id )  [static]
```

Fetching parking information from database using unique id.

Args:
    id: Parking's id.

Returns:
    A new Parking object.

5.13.3.5 get_parking_map()

```
list[list] classes.parking.Parking.get_parking_map (
              id )  [static]
```

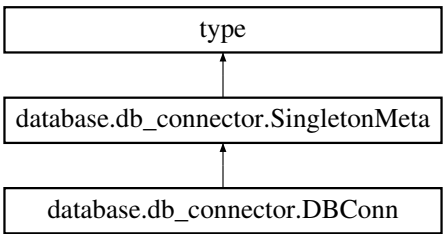
```
Creates a parking map as a matrix.
With places marked either as EMPTY or OCCUPIED.

Args:
    id:     Id of the parking.

Returns:
    A matrix with empty and occupied places with chargers on the parking.
```

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

- application/classes/parking.py

5.14 database.db_connector.SingletonMeta Class Reference

Inheritance diagram for database.db_connector.SingletonMeta:



Public Member Functions

- def __call__ (cls, *args, **kwargs)

Static Public Attributes

- **Lock**

5.14.1 Detailed Description

```
Thread-safe implementation of Singleton.
```

5.14.2 Member Function Documentation

5.14.2.1 __call__()

```
def database.db_connector.SingletonMeta.__call__ (
              cls,
             * args,
             ** kwargs )
```

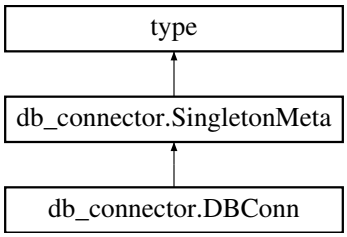
```
Possible changes to the value of the `__init__` argument do not affect
the returned instance.
```

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

- application/database/db_connector.py

5.15 db_connector.SingletonMeta Class Reference

Inheritance diagram for db_connector.SingletonMeta:



Public Member Functions

- def __call__ (cls, ∗args, ∗∗kwargs)

Static Public Attributes

- **Lock**

5.15.1 Detailed Description

```
Thread-safe implementation of Singleton.
```

5.15.2 Member Function Documentation

5.15.2.1 __call__()

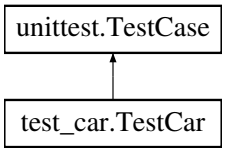
```
def db_connector.SingletonMeta.__call__ (
              cls,
             * args,
             ** kwargs )


Possible changes to the value of the `__init__` argument do not affect
the returned instance.
```

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

- algorithm/database/db_connector.py

5.16 test_car.TestCar Class Reference

Inheritance diagram for test_car.TestCar:



Public Member Functions

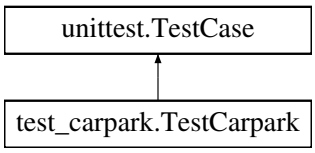
- def **test_create_car** (self)
- def **test_add_car_to_db** (self, patch_db_conn)
- def **test_get_car_from_db** (self, patch_db_cur)
- def **test_get_client_cars_from_db** (self, patch_db_cur, patch_get_car)
- def **test_is_parked_true** (self, patch_db_cur)
- def **test_is_parked_true** (self, patch_db_cur)
- def **test_car_park** (self, patch_db_conn)
- def **test_car_park_parked** (self)
- def **test_unpark_car** (self, patch_db_cur, patch_db_conn, patch_parked)
- def **test_unpark_unparked_car** (self, patch_parked)
- def **test_unpark_car** (self, patch_parked, patch_db_conn)
- def **test_change_dep_unparked** (self, patch_parked)

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

- application/tests/test_car.py

5.17 test_carpark.TestCarpark Class Reference

Inheritance diagram for test_carpark.TestCarpark:



Public Member Functions

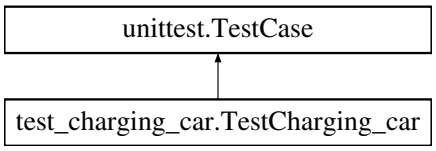
- def **test_create_carpark** (self)
- def **test_max_energy_use** (self)
- def **test_real_energy_use** (self)
- def **test_set_all_stations** (self)
- def **test_sort_cars** (self)

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

- algorithm/tests/test_carpark.py

5.18 test_charging_car.TestCharging_car Class Reference

Inheritance diagram for test_charging_car.TestCharging_car:



Public Member Functions

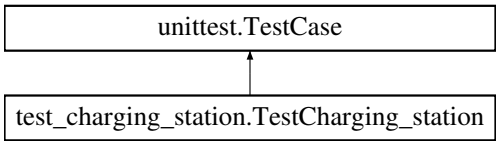
- def **test_create_car** (self)

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

- algorithm/tests/test_charging_car.py

5.19 test_charging_station.TestCharging_station Class Reference

Inheritance diagram for test_charging_station.TestCharging_station:



Public Member Functions

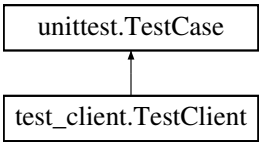
- def **test_create_station** (self)
- def **test_set_hour_to_go** (self)
- def **test_set_hour_to_go2** (self)
- def **test_set_below_start2** (self)
- def **test_set_below_start** (self)
- def **test_set_status** (self)
- def **test_set_status2** (self)
- def **test_set_status3** (self)
- def **test_set_status4** (self)
- def **test_set_status5** (self)
- def **test_max_energy_usage** (self)
- def **test_set_order** (self)
- def **test_set_order2** (self)
- def **test_set_order3** (self)
- def **test_energy_usage** (self)

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

- algorithm/tests/test_charging_station.py

5.20 test_client.TestClient Class Reference

Inheritance diagram for test_client.TestClient:



Public Member Functions

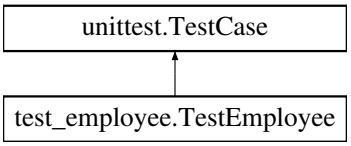
- def **test_create_client** (self)
- def **test_get_client_from_db** (self, patch_db_cur, patch_car)
- def **test_add_client_to_db** (self, patch_db_conn)
- def **test_add_client_car** (self, patch_car)
- def **test_save_client_cars** (self)
- def **test_save_client_cars_duplicates** (self)
- def **test_park_car** (self, patch_db_conn)
- def **test_park_parked_car** (self)
- def **test_unpark_car** (self, patch_db_cur, patch_db_conn, patch_parked)
- def **test_change_car_departure** (self, patch_car_change_dep)
- def **test_change_dep_unowned_car** (self)

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

- application/tests/test_client.py

5.21 test_employee.TestEmployee Class Reference

Inheritance diagram for test_employee.TestEmployee:



Public Member Functions

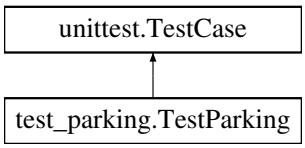
- def **test_create_employee** (self)
- def **test_get_employee_from_db** (self, patch_db_cur)

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

- application/tests/test_employee.py

5.22 test_parking.TestParking Class Reference

Inheritance diagram for test_parking.TestParking:



Public Member Functions

- def **test_create_parking** (self)
- def **test_get_parking_from_db** (self, patch_db_cur)
- def **test_get_employee_parking** (self, patch_account)
- def **test_get_all_parkings** (self, patch_db_cur)
- def **test_get_park_map** (self, patch_db_cur)

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

- application/tests/test_parking.py

5.23 to_database.To_database Class Reference

Static Public Member Functions

- def **insert_requests** (request, expenditure, carpark_id)
- def **get_curr_time** ()
- def **insert_new_charging** (stations)

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

- algorithm/database/to_database.py

Chapter 6

File Documentation

6.1 20221112_schema.ddl

```
1 DROP TABLE IF EXISTS requests CASCADE;
2 DROP TABLE IF EXISTS charging CASCADE;
3 DROP TABLE IF EXISTS chargers CASCADE;
4 DROP TABLE IF EXISTS cars CASCADE;
5 DROP TABLE IF EXISTS accounts CASCADE;
6 DROP TABLE IF EXISTS car_parks CASCADE;
7
8 CREATE TABLE accounts (
9     account_no          SERIAL NOT NULL,
10     name                VARCHAR(20) NOT NULL,
11     password            VARCHAR(30) NOT NULL,
12     email_address       VARCHAR(20) NOT NULL,
13     phone_no            CHAR(9) NOT NULL,
14     account_type        VARCHAR(8) NOT NULL,
15     cpa_car_park_id         INTEGER
16 );
17
18 ALTER TABLE accounts
19     ADD CONSTRAINT acc_arc CHECK ( ( account_type = 'EMPLOYEE' )
20
21                                    OR ( ( account_type = 'CLIENT')
22                                        AND
23                                        ( cpa_car_park_id IS NULL ) ) ) ;
24
25 ALTER TABLE accounts ADD CONSTRAINT acc_pk PRIMARY KEY ( account_no );
26
27 ALTER TABLE accounts ADD CONSTRAINT acc_name_un UNIQUE ( name );
28
29 CREATE INDEX acc_cpa__idx ON
30     accounts (
31         cpa_car_park_ID
32     ASC );
33
34
35 CREATE TABLE cars (
36     vin             VARCHAR(17) NOT NULL,
37     registration_no VARCHAR(9) NOT NULL,
38     model           VARCHAR(20) NOT NULL,
39     brand           VARCHAR(20) NOT NULL,
40     capacity        NUMERIC(6, 2) NOT NULL,
41     description     VARCHAR(100),
42     acc_account_no  INTEGER NOT NULL
43 );
44
45 ALTER TABLE cars ADD CONSTRAINT car_pk PRIMARY KEY ( vin );
46
47 ALTER TABLE cars ADD CONSTRAINT car_reg_no_un UNIQUE ( registration_no );
48
49 CREATE INDEX car_acc__idx ON
50     cars (
51         acc_account_no
52     ASC );
53
54
55 CREATE TABLE car_parks (
56     car_park_id SERIAL NOT NULL,
57     spaces_no   NUMERIC(3) NOT NULL,
58     city        VARCHAR(20) NOT NULL,
```

```
59     street      VARCHAR(20) NOT NULL,
60     building_no VARCHAR(6) NOT NULL
61 );
62
63 ALTER TABLE car_parks ADD CONSTRAINT cpa_pk PRIMARY KEY ( car_park_id );
64
65 CREATE TABLE chargers (
66     charger_code    CHAR(8) NOT NULL,
67     maximal_power   NUMERIC(3) NOT NULL,
68     charger_type    CHAR(2) NOT NULL,
69     description     VARCHAR(100),
70     cpa_car_park_id INTEGER NOT NULL
71 );
72
73 ALTER TABLE chargers
74     ADD CONSTRAINT cha_type_check CHECK ( ( charger_type = 'AC')
75                                                           OR
76                                                           ( charger_type = 'DC' ) );
77
78 ALTER TABLE chargers ADD CONSTRAINT cha_pk PRIMARY KEY ( charger_code );
79
80 CREATE INDEX cha_cpa__idx ON
81     chargers (
82         CPA_car_park_ID
83     ASC );
84
85
86 CREATE TABLE charging (
87     datetime          TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
88     base_charge_level NUMERIC(6, 2) NOT NULL,
89     charge_level      NUMERIC(6, 2) NOT NULL,
90     departure_datetime TIMESTAMP NOT NULL,
91     cha_charger_code  CHAR(8) NOT NULL,
92     car_vin           VARCHAR(17) NOT NULL
93 );
94
95 CREATE INDEX cin_cha__idxv1 ON
96     charging (
97         cha_charger_code
98     ASC );
99
100 ALTER TABLE charging
101     ADD CONSTRAINT cin_pk PRIMARY KEY ( car_vin,
102                                         datetime );
103
104
105 CREATE TABLE requests (
106     datetime        TIMESTAMP NOT NULL,
107     request         NUMERIC(4) NOT NULL,
108     expenditure     NUMERIC(4),
109     cpa_car_park_id INTEGER NOT NULL
110 );
111
112 CREATE INDEX req_cpa__idx ON
113     requests (
114         cpa_car_park_id
115     ASC );
116
117 ALTER TABLE requests ADD CONSTRAINT req_pk PRIMARY KEY ( datetime,
118                                                          cpa_car_park_id );
119
120 ALTER TABLE cars
121     ADD CONSTRAINT car_cli_fk FOREIGN KEY ( acc_account_no )
122         REFERENCES accounts ( account_no );
123
124 ALTER TABLE charging
125     ADD CONSTRAINT cha_car_fk FOREIGN KEY ( car_vin )
126         REFERENCES cars ( vin );
127
128 ALTER TABLE chargers
129     ADD CONSTRAINT cha_cpa_fk FOREIGN KEY ( cpa_car_park_id )
130         REFERENCES car_parks ( car_park_id );
131
132 ALTER TABLE charging
133     ADD CONSTRAINT cin_char_fk FOREIGN KEY ( cha_charger_code )
134         REFERENCES chargers ( charger_code );
135
136 ALTER TABLE accounts
137     ADD CONSTRAINT acc_cpa_fk FOREIGN KEY ( cpa_car_park_id )
138         REFERENCES car_parks ( car_park_id );
139
140 ALTER TABLE requests
141     ADD CONSTRAINT req_cpa_fk FOREIGN KEY ( cpa_car_park_id )
142         REFERENCES car_parks ( car_park_id );
143
144 CREATE OR REPLACE FUNCTION msgnontransferable()
145 RETURNS trigger AS
```

```
146 $$
147 BEGIN
148   RAISE EXCEPTION 'Cant change nontransferable value';
149 END;
150 $$
151 LANGUAGE plpgsql;
152
153 CREATE OR REPLACE FUNCTION car_account_changes()
154 RETURNS trigger AS
155 $$
156 BEGIN
157     PERFORM account_no FROM accounts WHERE account_no = NEW."acc_account_no" AND account_type =
      'EMPLOYEE';
158     IF FOUND THEN
159         RAISE EXCEPTION 'Employee account cant have assigned cars!';
160     ELSE
161         RETURN NEW;
162     END IF;
163 END;
164 $$
165 LANGUAGE plpgsql;
166
167 CREATE OR REPLACE TRIGGER fkntm_charging BEFORE
168     UPDATE OF cha_charger_code, car_vin ON charging
169 EXECUTE PROCEDURE msgnontransferable();
170
171 CREATE OR REPLACE TRIGGER fkntm_request BEFORE
172     UPDATE OF cpa_car_park_id ON requests
173 EXECUTE PROCEDURE msgnontransferable();
174
175 CREATE OR REPLACE TRIGGER employee_car_update BEFORE
176     INSERT OR UPDATE OF acc_account_no on CARS
177 FOR EACH ROW
178 EXECUTE PROCEDURE car_account_changes();
179
180 CREATE OR REPLACE VIEW clients_list AS
181 SELECT account_no, name, password, email_address, phone_no
182 FROM accounts
183 WHERE account_type LIKE 'CLIENT';
184
185 CREATE OR REPLACE VIEW parked_cars AS
186 SELECT vin, registration_no, model, brand, capacity, description, acc_account_no
187 FROM cars CAR JOIN charging CIN on CAR.vin = CIN.CAR_vin
188 WHERE cin.departure_datetime > CURRENT_TIMESTAMP;
189
190 CREATE OR REPLACE VIEW cars_charging AS
191 WITH currently_charged AS (
192     SELECT DISTINCT ON (car_vin)
193     base_charge_level, charge_level, car_vin, cha_charger_code, datetime, departure_datetime
194     FROM charging
195     WHERE CURRENT_TIMESTAMP < departure_datetime
196     ORDER BY car_vin
197 )
198 SELECT CIN.charge_level, CIN.base_charge_level, CIN.datetime, CIN.departure_datetime, CARS.capacity,
      CHA.Maximal_power, CHA.charger_type, CHA.charger_code, CPA.car_park_ID, CARS.vin
199 FROM (((car_parks as CPA join chargers AS CHA on CPA.car_park_id = CHA.cpa_car_park_id) JOIN
      currently_charged AS CIN ON CHA.charger_code = CIN.cha_charger_code)
200 JOIN cars ON cars.vin = CIN.car_vin);
201
202 INSERT INTO car_parks (spaces_no, city, street, building_no) VALUES
203 (15, 'Warszawa', 'Glowna', '15'),
204 (20, 'Gdansk', 'Posrednia', '22a'),
205 (10, 'Warszawa', 'Dluga', '4');
206
207 INSERT INTO accounts (name, password, email_address, phone_no, account_type)
208 VALUES ('gregory', 'haslo123', 'gregory@gmail.com', '991888777', 'CLIENT'),
209 ('barian', 'haslo123', 'barian@gmail.com', '992888777', 'CLIENT'),
210 ('carian', 'haslo123', 'carian@gmail.com', '993888777', 'CLIENT'),
211 ('darian', 'haslo123', 'darian@gmail.com', '994888777', 'CLIENT'),
212 ('marian', 'haslo123', 'marian@gmail.com', '995888777', 'CLIENT');
213
214 INSERT INTO accounts (name, password, email_address, phone_no, account_type, cpa_car_park_id)
215 VALUES ('employee', 'haslo123', 'mail@mail.com', '888666777', 'EMPLOYEE', 1),
216 ('a.bonk', 'haslo123', 'bonk@gmail.com', '818656777', 'EMPLOYEE', 1),
217 ('b.donk', 'haslo123', 'donk@gmail.com', '828666777', 'EMPLOYEE', 1),
218 ('c.wonk', 'haslo123', 'wonk@gmail.com', '838676777', 'EMPLOYEE', 2),
219 ('d.gonk', 'haslo123', 'gonk@gmail.com', '848686777', 'EMPLOYEE', 2),
220 ('e.monk', 'haslo123', 'monk@gmail.com', '858696777', 'EMPLOYEE', 3);
221
222 INSERT INTO chargers (charger_code, maximal_power, charger_type, description, cpa_car_park_id)
223 VALUES
224  ('01-00-00', 200, 'DC', 'Nice charger', 1),
225  ('01-00-01', 200, 'DC', 'Nice charger', 1),
226  ('01-00-02', 200, 'DC', 'Nice charger', 1),
227  ('01-00-03', 200, 'DC', 'Nice charger', 1),
228  ('01-00-04', 200, 'DC', 'Nice charger', 1),
229  ('01-00-05', 200, 'DC', 'Nice charger', 1),
```

```
230  ('01-00-06', 200, 'DC', 'Nice charger', 1),
231  ('01-00-07', 200, 'DC', 'Nice charger', 1),
232  ('01-00-08', 200, 'DC', 'Nice charger', 1),
233  ('01-00-09', 200, 'DC', 'Nice charger', 1),
234  ('01-01-00', 200, 'DC', 'Nice charger', 1),
235  ('01-01-01', 200, 'DC', 'Nice charger', 1),
236  ('01-01-02', 200, 'DC', 'Nice charger', 1),
237  ('01-01-03', 200, 'DC', 'Nice charger', 1),
238  ('01-01-04', 200, 'DC', 'Nice charger', 1),
239  ('02-00-00', 150, 'AC', 'Nice charger', 2),
240  ('02-00-01', 150, 'AC', 'Nice charger', 2),
241  ('02-00-02', 150, 'AC', 'Nice charger', 2),
242  ('02-00-03', 150, 'AC', 'Nice charger', 2),
243  ('02-00-04', 150, 'AC', 'Nice charger', 2),
244  ('02-00-05', 150, 'AC', 'Nice charger', 2),
245  ('02-00-06', 150, 'AC', 'Nice charger', 2),
246  ('02-00-07', 150, 'AC', 'Nice charger', 2),
247  ('02-00-08', 150, 'AC', 'Nice charger', 2),
248  ('02-00-09', 150, 'AC', 'Nice charger', 2),
249  ('02-01-00', 150, 'AC', 'Nice charger', 2),
250  ('02-01-01', 150, 'AC', 'Nice charger', 2),
251  ('02-01-02', 150, 'AC', 'Nice charger', 2),
252  ('02-01-03', 150, 'AC', 'Nice charger', 2),
253  ('02-01-04', 150, 'AC', 'Nice charger', 2),
254  ('02-01-05', 150, 'AC', 'Nice charger', 2),
255  ('02-01-06', 150, 'AC', 'Nice charger', 2),
256  ('02-01-07', 150, 'AC', 'Nice charger', 2),
257  ('02-01-08', 150, 'AC', 'Nice charger', 2),
258  ('02-01-09', 150, 'AC', 'Nice charger', 2),
259  ('03-00-00', 250, 'DC', 'Nice charger', 3),
260  ('03-00-01', 250, 'DC', 'Nice charger', 3),
261  ('03-00-02', 250, 'DC', 'Nice charger', 3),
262  ('03-00-03', 250, 'DC', 'Nice charger', 3),
263  ('03-00-04', 250, 'DC', 'Nice charger', 3),
264  ('03-01-00', 200, 'AC', 'Nice charger', 3),
265  ('03-01-01', 200, 'AC', 'Nice charger', 3),
266  ('03-01-02', 200, 'AC', 'Nice charger', 3),
267  ('03-01-03', 200, 'AC', 'Nice charger', 3),
268  ('03-01-04', 200, 'AC', 'Nice charger', 3);
269
270 INSERT INTO cars (vin, registration_no, model, brand, capacity, description, acc_account_no) VALUES
271 ('8AGW25JT38KC66775', 'FWS2936', 'R1S', 'Rivian', 2000.20, 'That is some crazy description', 1),
272 ('TMBWUTC46MZ3J9161', 'RP59547', 'R1T', 'Rivian', 2000.20, 'That is some crazy description', 2),
273 ('8BCTTKUX8JX8L4378', 'NKE9095', 'Taycan Cross Turismo', 'Porsche', 2000.20, 'That is some crazy
      description', 3),
274 ('2TNH62B71YD1W6237', 'PL91879', 'I-Pace', 'Jaguar', 2000.20, 'That is some crazy description', 3),
275 ('YS4PPFMR05D2E1952', 'SZA9020', 'e-tron Sportback', 'Audi', 2000.20, 'That is some crazy description',
      4),
276 ('WBAMPXN54ZLLL6978', 'WGR0691', 'e-tron', 'Audi', 2000.20, 'That is some crazy description', 5),
277 ('2DGRZV4Y52D4B5641', 'ESI8184', 'Taycan Sport Turismo', 'Porsche', 2000.20, 'That is some crazy
      description', 5),
278 ('3F1P667R76TZJ9662', 'DWR8470', 'e-tron GT', 'Audi', 2000.20, 'That is some crazy description', 5);
279
280 INSERT INTO charging (base_charge_level, charge_level, departure_datetime, cha_charger_code, car_vin)
      VALUES
281 (500, 500, (NOW() + interval '1 month'), '01-00-00', '8AGW25JT38KC66775'),
282 (500, 500, (NOW() + interval '1 month'), '01-00-01', 'TMBWUTC46MZ3J9161'),
283 (500, 500, (NOW() + interval '1 month'), '01-00-08', '8BCTTKUX8JX8L4378'),
284 (500, 500, (NOW() + interval '1 month'), '01-01-01', '2TNH62B71YD1W6237'),
285 (500, 500, (NOW() + interval '1 month'), '01-01-03', 'YS4PPFMR05D2E1952'),
286 (500, 500, (NOW() + interval '1 month'), '01-01-04', '2DGRZV4Y52D4B5641'),
287 (500, 500, (NOW() + interval '1 month'), '01-00-05', '3F1P667R76TZJ9662');
```

Index