Endpoint function: Authenticate user with username and password

Endpoint: /api/authenticate/default

python function: sql_related.authenticate_default(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
        "data":
        {
                "username": <user username (String)>,
                "password": <user password (String)>
        }
}
```

JSON Response Format (successful login):

Header: 200

```
{
        "data":
        {
                "id": <user database id (int)>,
                "name_first": <user first name (String)>,
                "name_last": <user last name (String)>,
                "sys_username": <user username (String)>,
                "email_google": <user google email (String)>,
                "email_fb": <user facebook email (String)>,
                “image_url”: <user image url (String)>,
                "phone_number": <user phone number (String / null)>,
                "sys_user_role":
                {
                        "id": <user role database id (int)>,
                        "name": <user role name (String)>
                }
        }
}
```

Response (failed login):

Header: “Failed Authentication” 401

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400

Endpoint function: Authenticate user with email via Google / Facebook

Endpoint:

- /api/authenticate/google (google authentication)
- /api/authenticate/facebook (facebook authentication) (Not final)

python function: sql_related.authenticate_email(database, data, google)

- database = main database object (db)
- data = JSON data directly from HTTP request
- google = boolean, true if google login, false if fb login

HTTP Request Type: POST

JSON Request Format:

```
{
        "data":
        {
                "email": <email>
        }
}
```

JSON Response Format (successful login):

Header: 200

```
{
        "data":
        {
                "id": <user database id (int)>,
                "name_first": <user first name (String)>,
                "name_last": <user last name (String)>,
                "sys_username": <user username (String)>,
                "email_google": <user google email (String)>,
                "email_fb": <user facebook email (String)>,
                “image_url”: <user image url (String)>,
                "phone_number": <user phone number (String / null)>,
                "sys_user_role":
                {
                        "id": <user role database id (int)>,
                        "name": <user role name (String)>
                }
        }
}
```

Response (failed login):

Header: “Failed Authentication” 401

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400

Endpoint function: Request all company_products and their stock

Endpoint:

- /api/inventory

python function: sql_related.request_company_product(database)

- database = main database object (db)

HTTP Request Type: GET

JSON Response Format:

Header: 200

```
{
        "data":
        [
                {
                        "id": <product database id (int)>,
                        "company":
                        {
                                "id": <company database id (int)>,
                                "name": <company name (String)>,
                                “image_url”: <user image url (String)>,
                                "zones":
                                [
                                    {
                                        "id": <zone database id (int)>,
                                        "name": <zone name (String)>
                                    },
                                     ... (Additional zones, same format)
                                ]
                        },
                        "name": <product name (String)>,
                        "price_buy": <product purchase price (Float)>,
                        "price_sell": <product sell price (Float)>,
                        "units_per_price": <amount of units per price_buy/sell (int)>,
                        "price_sell_per_unit": <product sell price per single unit (Float)>,
                        "stock": <amount of units in stock (int)>,
                        "description": <product description (String)>
                },
                 ... (Additional products, same format)
        ]
}
```

Endpoint function: Request all shop_orders that need to be delivered

Endpoint:

- /api/orders/not_delivered

python function: sql_related.request_shop_order_not_delivered(database)

- database = main database object (db)

HTTP Request Type: GET

JSON Response Format:

```
Header: 200
{
    "data":
    [
        {
            "id": <order database id (int)>,
            "shop":
```

```
{
        "id": <shop database id (int)>,
        "name": <shop name (String)>,
        "email": <shop name (String / null)>,
        “image_url”: <user image url (String)>,
        "phone_number": <shop phone number (String / null)>,
        "category": (may be null)
        {
                "id": <shop category database id (int)>,
                "type": <shop category type (String)>
        },
        "zones":
        [
                {
                        "id": <zone database id (int)>,
                        "name": <zone name (String)>
                },
                … (Additional zones, same format)
        ],
        "street": <shop street (String)>,
        "city": "<shop city(String)>,
        "providence": <shop providence (String)>,
        "zip_4": <shop zip+4 code (String)>
},
"price_due": <order price (Float)>,
"price_paid": <order was paid (Boolean)>,
“memo”: <order memo (String)>,
"date_ordered": <order placed date (String)>,
"date_delivered_projected": <order expected date (String)>,
"date_delivered": null,
"order_taker":
{
        "id": <user database id (int)>,
        "name_first": <user first name (String)>,
        "name_last": <user last name (String)>,
        "sys_username": <user username (String)>,
        "email_google": <user google email (String)>,
        "email_fb": <user facebook email (String)>,
        “image_url”: <user image url (String)>,
        "phone_number": <user phone number (String / null)>,
          "sys_user_role":
          {
                    "id": <user role database id (int)>,
                    "name": <user role name (String)>
          }
},
"order_fulfiller": null,
"completed": false
“shop_order_items”:
[
          {
                    company_product:
                    {
                              "id": <product database id (int)>,
                              "company":
                              {
                                        "id": <company database id (int)>,
                                        "name": <company name (String)>,
                                        “image_url”: <user image url (String)>,
                                        "zones":
                                        [
                                                  {
                                                            "id": <zone database id
                                                                      (int)>,
                                                            "name": <zone name
                                                                      (String)>
                                                  },
                                                  … (Additional zones, same
                                                            format)
                                        ]
                              },
                              "name": <product name (String)>,
                              "price_buy": <product purchase price
                                        (Float)>,
                              "price_sell": <product sell price (Float)>,
                                        "units_per_price": <amount of units
                                                  per price_buy/sell (int)>,
                              "price_sell_per_unit": <product sell price per
                                        single unit (Float)>,
                              "description": <product description (String)>
```

```
                                        },
                                        quantity_units: <units of company product ordered (int)>
                              },
                              … (Additional order items)
                    ]
          },
          … (Additional shops, same format)
     ]
}
```

Endpoint function: Request all shop_orders scheduled for today

Endpoint:

- /api/orders/today

python function: sql_related.request_shop_order_today(database)

- database = main database object (db)

HTTP Request Type: GET

JSON Response Format:

```
Header: 200
{
     "data":
     [
          {
               "id": <order database id (int)>,
               "shop":
               {
                    "id": <shop database id (int)>,
                    "name": <shop name (String)>,
                    "email": <shop name (String / null)>,
                    “image_url”: <user image url (String)>,
                    "phone_number": <shop phone number (String / null)>,
                    "category": (may be null)
                    {
                         "id": <shop category database id (int)>,
                         "type": <shop category type (String)>
                    },
                    "zones":
```

```
        [
            {
                "id": <zone database id (int)>,
                "name": <zone name (String)>
            },
            … (Additional zones, same format)
        ],
        "street": <shop street (String)>,
        "city": "<shop city(String)>,
        "providence": <shop providence (String)>,
        "zip_4": <shop zip+4 code (String)>
    },
    "price_due": <order price (Float)>,
    "price_paid": <order was paid (Boolean)>,
    “memo”: <order memo (String)>,
    "date_ordered": <order placed date (String)>,
    "date_delivered_projected": <order expected date (String)>,
    "date_delivered": null,
    "order_taker":
    {
        "id": <user database id (int)>,
        "name_first": <user first name (String)>,
        "name_last": <user last name (String)>,
        "sys_username": <user username (String)>,
        "email_google": <user google email (String)>,
        "email_fb": <user facebook email (String)>,
        “image_url”: <user image url (String)>,
        "phone_number": <user phone number (String / null)>,
        "sys_user_role":
        {
            "id": <user role database id (int)>,
            "name": <user role name (String)>
        }
    },
    "order_fulfiller": null,
    "completed": false
    “shop_order_items”:
    [
        {
            company_product:
                    {
                        "id": <product database id (int)>,
                        "company":
                        {
                            "id": <company database id (int)>,
                            "name": <company name (String)>,
                            “image_url”: <user image url (String)>,
                            "zones":
                            [
                                {
                                    "id": <zone database id
                                        (int)>,
                                    "name": <zone name
                                        (String)>
                                },
                                … (Additional zones, same
                                    format)
                            ]
                        },
                        "name": <product name (String)>,
                        "price_buy": <product purchase price
                            (Float)>,
                        "price_sell": <product sell price (Float)>,
                            "units_per_price": <amount of units
                                per price_buy/sell (int)>,
                        "price_sell_per_unit": <product sell price per
                            single unit (Float)>,
                        "description": <product description (String)>
                    },
                    quantity_units: <units of company product ordered (int)>
                },
                … (Additional order items)
            ]
        },
        … (Additional shops, same format)
    ]
}
```

Endpoint function: Request all shops

Endpoint:

- /api/shops/all

python function: sql_related.request_shop(database)

- database = main database object (db)

HTTP Request Type: GET

JSON Response Format:

```
Header: 200
{
	"data":
	[
		{
			"id": <shop database id (int)>,
			"name": <shop name (String)>,
			"email": <shop name (String / null)>,
			“image_url”: <user image url (String)>,
			"phone_number": <shop phone number (String / null)>,
			"category": (may be null)
			{
				"id": <shop category database id (int)>,
				"type": <shop category type (String)>
			},
			"zones":
			[
				{
					"id": <zone database id (int)>,
					"name": <zone name (String)>
				},
				… (Additional zones, same format)
			],
			"street": <shop street (String)>,
			"city": "<shop city(String)>,
			"providence": <shop providence (String)>,
			"zip_4": <shop zip+4 code (String)>
		},
		… (Additional shops, same format)
	]
}
```

Endpoint function: Request all zones

Endpoint:

- /api/zones/all

python function: sql_related.request_zone(database)

- database = main database object (db)

HTTP Request Type: GET

JSON Response Format:

```
{
          "data":
          [
                    {
                              "id": <zone database id (int)>,
                              "name": <zone name (String)>
                    },
                    … (Additional zones, same format)
          ]
}
```

Endpoint function: Request all shop_categories

Endpoint:

- /api/shop_categories/all

python function: sql_related.request_shop_category(database)

- database = main database object (db)

HTTP Request Type: GET

JSON Response Format:

```
{
          "data":
          [
                    {
                              "id": <shop category database id (int)>,
                "type": <shop category type (String)>
            },
            … (Additional shop categories, same format)
        ]
}
```

Endpoint function: Create a new shop

Endpoint: /api/create/shop

python function: sql_related.create_shop(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
    "data":
    {
        "name": <shop name (String)>,
        "email": <shop email (String / null)>,
        “image_url”: <user image url (String)>,
        "phone_number": <shop phone number(String / null)>,
        "category": <shop category id (int / null)>,
        "zones":
        [
            {
                "id": <zone id (int)>
            },
            … (Additional zones, same format)
        ],
        "street": <shop street (String)>,
        "city": <shop city (String)>,
        "providence": <shop providence (String)>,
        "zip_4": <shop zip+4 code (String)>
    }
}
```

JSON Response Format:

Header: 200

```
{
	"data":
	{
		"id": <shop database id (int)>,
		"name": <shop name (String)>,
		"email": <shop name (String / null)>,
		“image_url”: <user image url (String)>,
		"phone_number": <shop phone number (String / null)>,
		"category": (may be null)
		{
			"id": <shop category database id (int)>,
			"type": <shop category type (String)>
		},
		"zones":
		[
			{
				"id": <zone database id (int)>,
				"name": <zone name (String)>
			},
			… (Additional zones, same format)
		],
		"street": <shop street (String)>,
		"city": "<shop city(String)>,
		"providence": <shop providence (String)>,
		"zip_4": <shop zip+4 code (String)>
	}
}
```

Response (bad zone id):

Header: “Zone does not exists with id provided” 400

Response (bad shop category id):

Header: “Shop category does not exist with id provided” 400

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400

Endpoint function: Create a new zone

Endpoint: /api/create/zone

python function: sql_related.create_zone(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
        "data":
        {
                "name": <zone name (String)>
        }
}
```

JSON Response Format:

Header: 200

```
{
        "data":
        {
                "id": <zone database id (int)>,
                "name": <zone name (String)>
        }
}
```

Response (bad zone name):

Header: "Zone already exists with that name" 400

Response (bad JSON structure / typing):

Header: "<detailed error as to what is wrong>" 400

Endpoint function: Create a shop_category

Endpoint: /api/create/shop_category

python function: sql_related.create_shop_category(database, data)

- database = main database object (db)

- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:
```
{
        "data":
        {
                "type": <shop category type (String)>
        }
}
```

JSON Response Format:
Header: 200
```
{
        "data":
        {
                "id": <shop category database id (int)>,
                "type": <shop category type (String)>
        }
}
```

Response (bad shop category name):
Header: “Shop category already exists with that name” 400

Response (bad JSON structure / typing):
Header: “<detailed error as to what is wrong>” 400

Endpoint function: Create a shop_order

Endpoint: /api/create/shop_order

python function: sql_related.create_shop_order(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
	"data":
	{
		"shop_id": <shop database id (int)>,
		"price_paid": <if price was paid at time of ordering (Boolean)>,
		“deliver_days_from_today”: <days from today that order will be delivered (int)>,
		“memo”: <order memo (String)>,
		"order_taker_id": <order taker user database id (int)>,
		"order_items":
		[
			{
				"id": <company product database id (int)>,
				"quantity_units": <quantity of item ordered (int)>
			},
			… (Additional order items, same format)
		]
	}
}
```

JSON Response Format:

Header: 200

```
{
	"data":
	{
		"id": <order database id (int)>,
		"shop":
		{
			"id": <shop database id (int)>,
			"name": <shop name (String)>,
			"email": <shop name (String / null)>,
			“image_url”: <user image url (String)>,
			"phone_number": <shop phone number (String / null)>,
			"category": (may be null)
			{
				"id": <shop category database id (int)>,
				"type": <shop category type (String)>
			},
			"zones":
			[
        {
            "id": <zone database id (int)>,
            "name": <zone name (String)>
        },
        … (Additional zones, same format)
    ],
    "street": <shop street (String)>,
    "city": "<shop city(String)>,
    "providence": <shop providence (String)>,
    "zip_4": <shop zip+4 code (String)>
},
"price_due": <order price (Float)>,
"price_paid": <order was paid (Boolean)>,
“memo”: <order memo (String)>,
"date_ordered": <order placed date (String)>,
"date_delivered_projected": <order expected date (String)>,
"date_delivered": null,
"order_taker":
{
    "id": <user database id (int)>,
    "name_first": <user first name (String)>,
    "name_last": <user last name (String)>,
    "sys_username": <user username (String)>,
    "email_google": <user google email (String)>,
    "email_fb": <user facebook email (String)>,
    “image_url”: <user image url (String)>,
    "phone_number": <user phone number (String / null)>,
    "sys_user_role":
    {
        "id": <user role database id (int)>,
        "name": <user role name (String)>
    }
},
"order_fulfiller": null,
"completed": false
“shop_order_items”:
[
    {
        company_product:
        {
```

```
                    "id": <product database id (int)>,
                    "company":
                    {
                        "id": <company database id (int)>,
                        "name": <company name (String)>,
                        “image_url”: <user image url (String)>,
                        "zones":
                        [
                            {
                                "id": <zone database id (int)>,
                                "name": <zone name (String)>
                            },
                            … (Additional zones, same format)
                        ]
                    },
                    "name": <product name (String)>,
                    "price_buy": <product purchase price (Float)>,
                    "price_sell": <product sell price (Float)>,
                    "units_per_price": <amount of units per price_buy/sell (int)>,
                    "price_sell_per_unit": <product sell price per single unit
                        (Float)>,
                    "description": <product description (String)>
                },
                quantity_units: <units of company product ordered (int)>
            },
            … (Additional order items)
        ]
    }
}
```

Response (bad deliver_days_from_today):

Header: “Invalid delivery date, must be at least one day from today” 400

Response (bad company product id):

Header: “Company product does not exist with id provided” 400

Response (bad order taker id):

Header: “Order Taker does not exist with id provided” 400

Response (bad shop id):

Header: “Shop does not exist with id provided” 400

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400

Endpoint function: Update a shop_order as delivered

Endpoint: /api/deliver/shop_order

python function: sql_related.update_shop_order_delivered(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
        "data":
        {
                "shop_order_id": <shop order database id (int)>,
                "order_fulfiller_id": <order fulfiller user database id (int)>
        }
}
```

JSON Response Format:

Header: 200

```
{
        "data":
        {
                "request_payment": <request payment for order (Boolean)>
        }
}
```

Response (shop order already completed):

Header: “Shop order is already completed” 400

Response (bad shop order id):

Header: “Shop order does not exist with id provided” 400

Response (bad order fulfiller id):

Header: “Order Fulfiller does not exist with id provided” 400

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400

Endpoint function: Get order taker goal data for current month

Endpoint: /api/goal/order_taker

python function: sql_related.goal_order_taker(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
        "data":
        {
                "order_taker_id": <order taker user database id (int)>
        }
}
```

JSON Response Format:

Header: 200

```
{
        "data":
        {
                {
                        "num_orders_total": <number of orders placed by order taker (int)>,
                        "current_value_total": <money value of orders placed by order taker (Float)>,
                        "goal_total": <money value of order taker’s goal (Float)>,
                        "orders_paid":
                        {
                                "num_orders": <number of orders that have been paid for (int)>,
                                "current_value": <money value of orders that have been paid for (Float)>
                        },
                        "orders_pending":
```

```
                    {
                              "num_orders": <number of orders that have not been paid for (int)>,
                              "current_value": <money value of orders that have not been paid for
                                        (Float)>
                    }
          }

     }
}
```

Response (bad user id):

Header: “Invalid user id” 400

Response (bad order taker id):

Header: “User is not an order taker” 400

Response (no goal data for current month):

Header: “No goal data for this month found for order taker” 404

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400

Endpoint function: Create order taker goal data for current month

Endpoint: /api/goal/order_taker/new

python function: sql_related.goal_order_taker_new(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
     "data":
     {
          "order_taker_id": <order taker user database id (int)>
          "goal_total": <goal money value (int / Float)>
     }
```

}

JSON Response Format:

Header: 200

```
{
	"data":
	{
		"order_taker": <order taker user database id (int)>,
		"month":  <current month (int)>,
		"year": <current year (int)>,
		"goal_value": <goal money value (int / Float)>
	}
}
```

Response (bad user id):

Header: “Invalid user id” 400

Response (bad order taker id):

Header: “User is not an order taker” 400

Response (goal data already exists for current month):

Header: “Goal for order taker already exists for this month” 404

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400

Endpoint function: Update inventory stock

Endpoint:

- /api/inventory/update

python function: sql_related.inventory_update(database, data)

- database = main database object (db)
- data = JSON data directly from HTTP request

HTTP Request Type: POST

JSON Request Format:

```
{
        "data":
        [
                {
                        "company_product_id": <company_product database id (int)>
                        "stock_delta": <value the stock should change by (int)>
                }
                … (Additional items, same format)
        ]
}
```

JSON Response Format:

Header: 200

```
{
        "data":
        [
                {
                        "id": <product database id (int)>,
                        "company":
                        {
                                "id": <company database id (int)>,
                                "name": <company name (String)>,
                                “image_url”: <user image url (String)>,
                                "zones":
                                [
                                        {
                                                "id": <zone database id (int)>,
                                                "name": <zone name (String)>
                                        },
                                         … (Additional zones, same format)
                                ]
                        },
                        "name": <product name (String)>,
                        "price_buy": <product purchase price (Float)>,
                        "price_sell": <product sell price (Float)>,
                        "units_per_price": <amount of units per price_buy/sell (int)>,
                        "price_sell_per_unit": <product sell price per single unit (Float)>,
                        "stock": <amount of units in stock (int)>,
                        "description": <product description (String)>
                },
                    … (Additional products, same format)
          ]
}
```

Response (bad company product id):

Header: “Invalid company product id” 400

Response (bad JSON structure / typing):

Header: “<detailed error as to what is wrong>” 400