|  |  |  |
| --- | --- | --- |
| **Program** | **Week** | **products** |
| id | Program\_id | Program\_id |
| semestr\_no | Week\_no | type |
| school\_year | Start\_data | min\_amount |
| fruitVeg\_price | End\_data | is\_fruitVeg |
| Dairy\_price |  | Product\_id |
| Start\_date |  |
| End\_date |  |
| dairy\_min\_in\_week |  |
| fruitVeg\_min\_in\_week |  |
| dairy\_amount |  |
| fruitVeg\_amount |  |

|  |  |  |
| --- | --- | --- |
| **School** | **Contract** | **Record** |
| Id | school\_id | Id |
| name | contract\_no | Contract\_id |
| address | contract\_year | current\_date |
| City | program\_id | Product\_id |
| Nip | Contract\_id | state (def = prepared, printed, notEnoughtAmount,  notDelivered, delivered ) |
| regon | validity\_date |  |
| email | fruitveg\_products |  |
| responsible\_person | dairy\_products |  |
|  | is\_annex (def=fasle) |  |
|  | is\_latest (def = true) |  |

CREATE TABLE `SCHOOL` (

`id` INTEGER,

`nick` TEXT NOT NULL UNIQUE,

`name` TEXT NOT NULL,

`address` TEXT NOT NULL,

`city` TEXT NOT NULL,

`nip` TEXT UNIQUE,

`regon` TEXT NOT NULL UNIQUE,

`email` TEXT NOT NULL,

`responsible\_person` TEXT,

`representant` TEXT,

`representant\_nip` TEXT,

`representant\_regon` TEXT,

`phone` TEXT NOT NULL,

PRIMARY KEY(`id`)

);

|  |  |  |  |
| --- | --- | --- | --- |
| **Order** | **Storage** | **Deliver** | **Driver** |
| Order\_date | Product\_id | dirver\_id | id |
| Product\_id | amount | Record\_id | name |
| Supplied (def= false) |  |  |
|  |  |  |
| Amount (in packages or foreseen packages) |  |  |

Supplied=true => add to amount in storage

Add record => decrease storage amount and state = prepared (gui color yellow) if the product is available in storage, if not available create and state = notEnoughAmount (gui color grey), finished (gui color green)

See the lacks in storage from notEnoughAmount state – possibly crate order from it and print order (to know how many packages have to be made)

After storage is increased (notEnoughtAmount obseravable on storage) -> change this to prepared and enable printing; possibly remove records with prepared or notEnoughtAmount state

Drag and drop list or sth like this with prepared records -> add to driver

Update storage -> after counting the real storage

|  |
| --- |
| **Expences** |
| id |
| Date |
| Description |
| Amount |
| Type (paycheck, product payment, other) |

Zielona Góra-> Zielonej Górze (lista za mapowaniem)

Funkcjonalnosci:

1. Uzupelnianie docxow: wz’tek, aneksow, oswiadczen, ewidencji, wniosku o pomoc i umow
2. Edytowanie danych szkol
3. Dodanie nowego programu (podstawowe dane, cena za paczke, ilość tygodnie, ilość podan itp.)
4. Generowanie wz’tek, wywozow dla kierowcow na podstawie WZ (recordow)
5. Przygotowanie zamówienia (edycja zamówienia), stan magazynowy
6. Logowanie: inny panel dla mamy, Mai (paczkowy), dla taty (dodawanie wydatków na paczki, wydatki gielda, wyplaty, samochody)
7. Rejestr <na jaki program> : dane szkoły, numer umowy, liczba dzieci na warzywa i mleko, informacje o aneksie czy był i na ile dzieci (wszystkie aneksy musza być)

Python logging module -> exceptions etc.

Ngrok -> żeby był dostep po URL a tata nie musial dostawac się po zdalnym dospie