Food ordering

Semestrálny projekt z ASwS 2023

Bc. Urbanek Rastislav, FMFI UK, 8.6.2023

2023

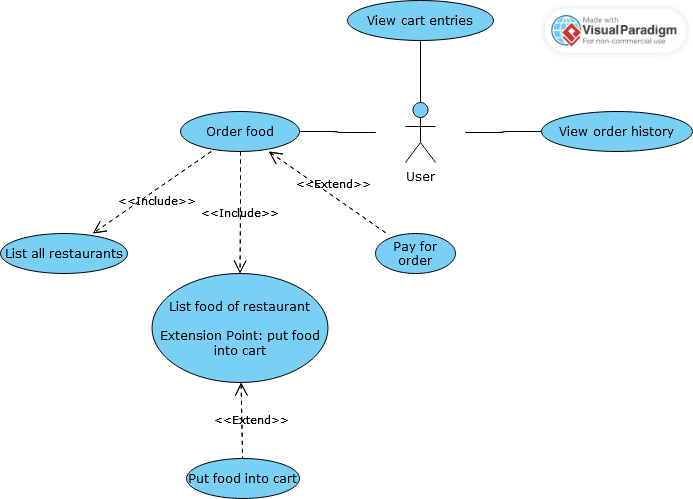
# Úvod

Aplikácia slúži ako trenažér na .NET a architektonické štýly. Jedná sa o jednoduchú CLI aplikáciu pripojenú na Postgresql databázu. Aplikácia umožňuje používateľovi vyberať si jedlá z obchodov, tvoriť objednávky a prezerať históriu objednávok

# Prípady použitia

## Diagram

Pay for order je jebnuty a nedal som extension point (ked tak moze byt ze pay on sight)



## Scenáre

POZN: MA BYT PRE A POST CONDITION

Aktér: Používateľ

Popis: Prezretie košíka

Tok scenára:

1. Používateľ si v menu zvolí príslušný príkaz
2. Systém vypíše obsah košíka
   1. Systém oznámi, že obsah košíka je prázdny

Aktér: Používateľ

Popis: Prezretie histórie objednávok

Tok scenára:

1. Používateľ si v menu zvolí príslušný príkaz
2. Systém vyzve používateľa aby vybral, ktorú objednávku si chce prezrieť
3. Používateľ zvolí objednávku
   1. Systém oznámi, že bola zvolená neexistujúca objednávka
4. Systém vypíše obsah objednávky

Aktér: Používateľ

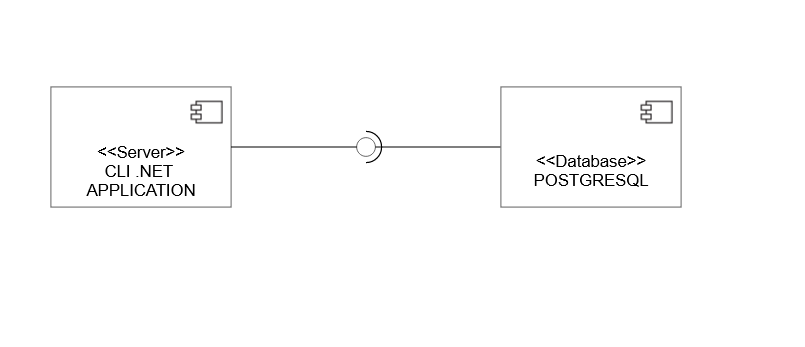
Popis: Objednanie jedla

Tok scenára:

1. Používateľ si v menu zvolí príslušný príkaz
2. Systém vyzve používateľa aby vybral z ktorej reštaurácie jedlo chce
3. Používateľ zadá reštauráciu
   1. Systém oznámi, že bola zvolená neexistujúca reštaurácia
   2. Systém oznámi, že bola zvolená reštaurácia ktorá už má zatvorené (?????)
4. Systém vyzve používateľa, aby si vybral jedlo z ponuky reštaurácie
5. Používateľ zadá jedlo
   1. Systém oznámi, že bola zvolená neexistujúca jedlo
6. Systém vyzve používateľa k platbe
7. Používateľ zaplatí
   1. Používateľ nezaplatí, no môže neskôr

# Architektúra systému

CLI klient komunikujúci cez EF kontext a db drajver s databázou

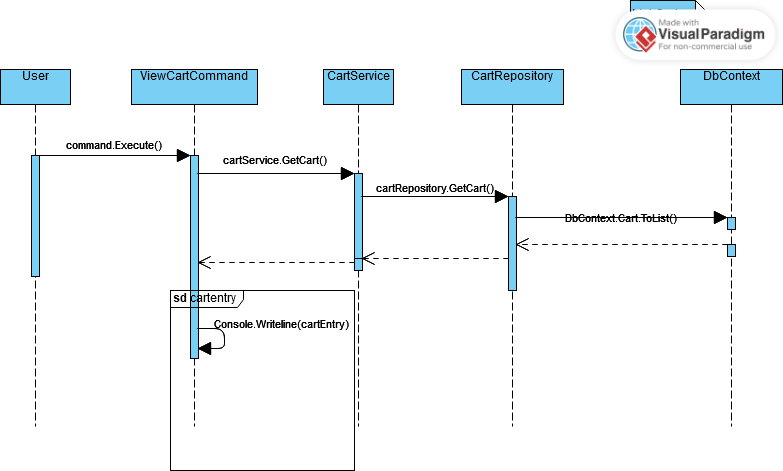


# Analýza

Mali by ste analyzovať doménu, pôvodný informačný systém v nej, ak existuje, ale vy  
budete v dokumente uvádzať hneď novonavrhnuté efektívne scenáre a štruktúru systému  
v nasledovných diagramoch:

## Sequence Diagram

(min. 2x), popisujúce scenár pripadov použitia pomocou interakcií medzi  
objektami, identifikujeme tak potrebné objekty-triedy a ich metódy, nezabudnúť na Loop, Alt, Opt a ich constrainty



## Activity diagram

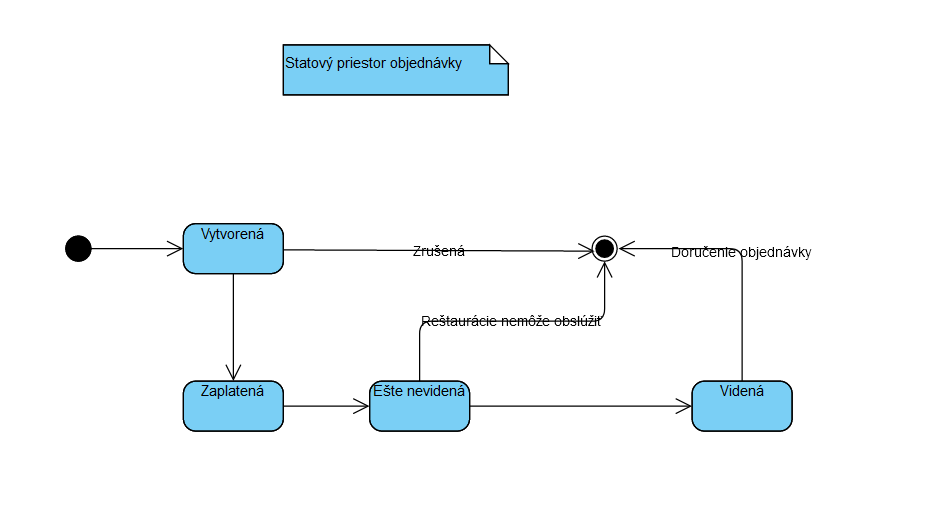
## Activity diagram

(min. jeden) pre vybraný scenár alebo algoritmus. Nemiešať elementy decision-  
merge (pre alternatívne vetvy) a fork-join (pre paralelné vetvy). Použite swimlanes, ale nie paušálne pre celý systém, ale pre jednotlivé objekty/triedy

## State diagram

Pozn: MAJU BYT OPERACIE, nedaj boze do, onentry/exit

(min. jeden). Uviesť pre ktorú konkrétnu triedu modelujeme stavový priestor a  
uviesť jej metódy. V samotnom diagrame nezabudnúť udalosti/podmienky na hranách, operácie/metódy triedy najmä v stavoch, menej na hranách



# Návrh a implementácia, identifikácia vzorov

## Class Diagram

odvodený od scenárov (SqD a AcD), potrebných objektov a metód. Pri  
agregácii a dedení nie je potrebné pomenovať väzby, pri klasickej asociácii uviesť názov vzťahu a kardinalitu, prípadne role objektov a pod.; pre atribúty môžete typ, pri metódach môžete zoznam parametrov, ich typ, návratovú hodnotu, nemusíte. Bude to jeho výsledná forma aj s identifikovanými vzormi ktoré budú identifikované v ofarbených poznámkach pri triedach (rola:vzor).

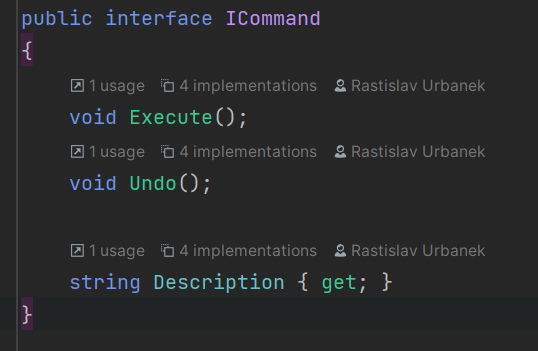
## model (class diagram) implemenovaného vzoru,

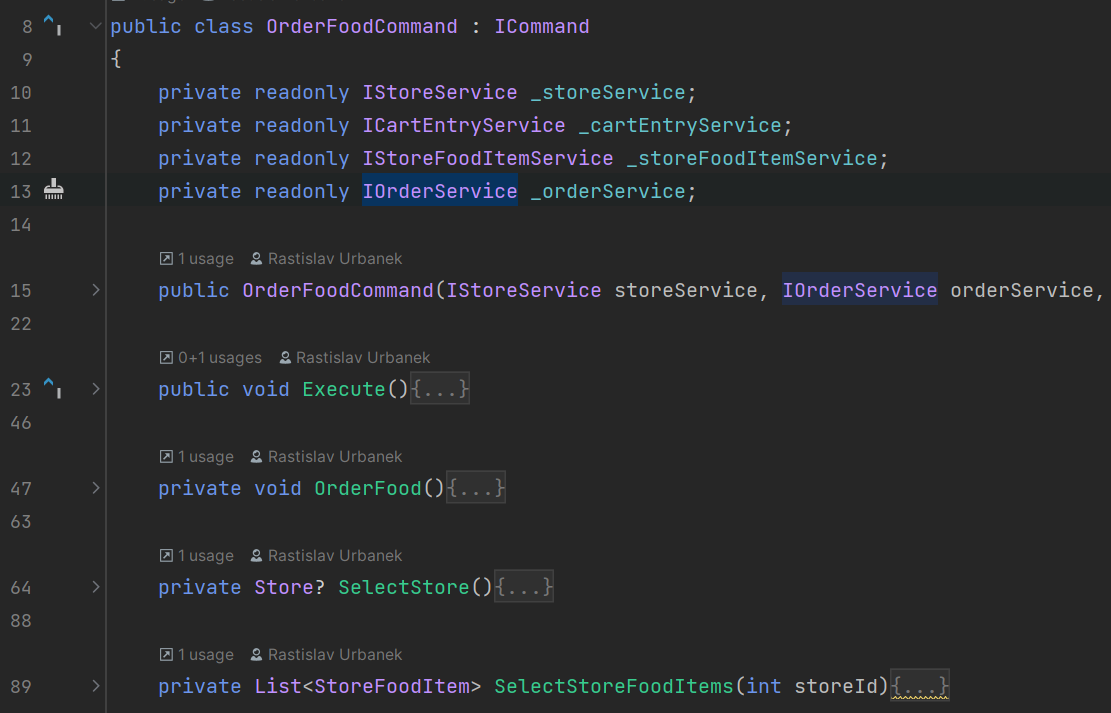
kde triedy majú dôležité metódy a atribúty +  
krátky popis + fragment zdrojáku vzoru z vašej jednoduchej funkčnej aplikácie aspoň v cmdline,

POZN: ze ma rad behavioralne? Nemam modely

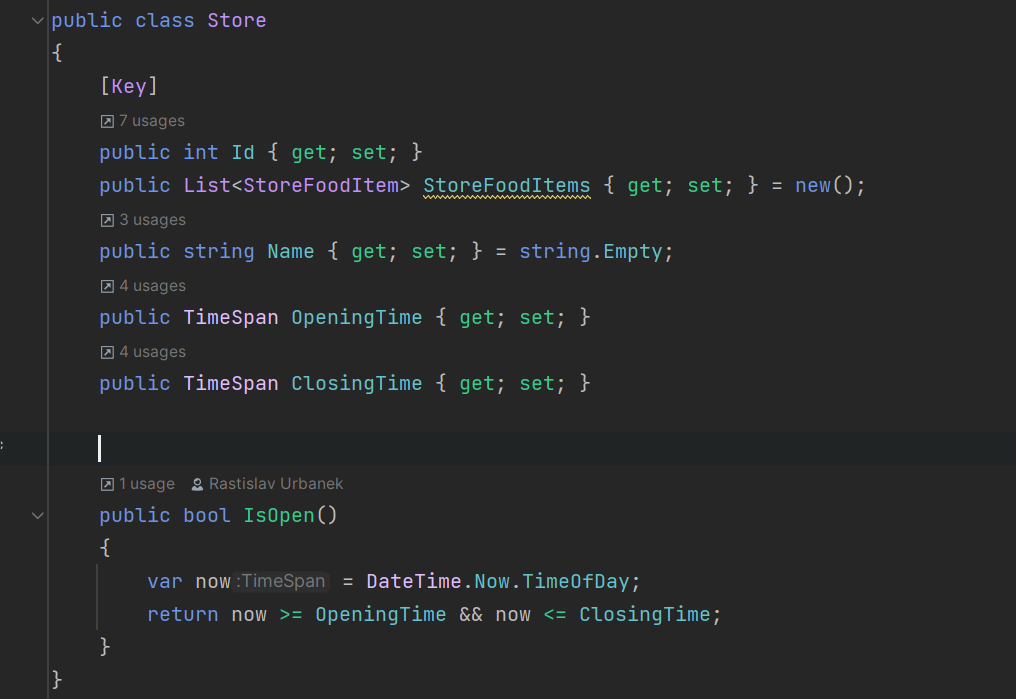
COMMAND

Kto je invoker, kto je receiver



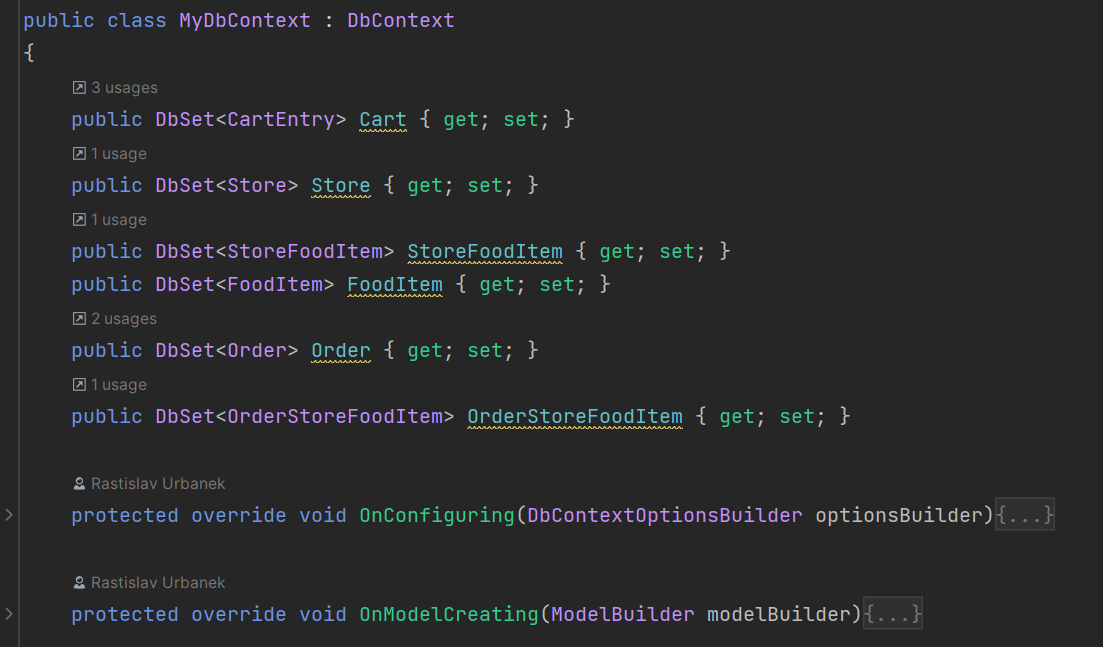


DTO



Context Object

Pozn. he seemed OK with this



Service



## implementácia vzorov v scenároch prípadov použitia: skopírujete SqD + poznámky ako v 5.1.