Interface. Vzťahy medzi triedami. Návrhové vzory. Observer. Strategy. Singleton.

Organizačné

- Od 9.03.2026 bude potrebné pravidelne pracovať na projekte. 2 commity týždenne do repozitára v rámci classroomu.
- Ak ste do repozitára commitovali prácu na cvičeniach, **tak od 09.03.2026 ju tam už nemôžete mať.**
- Len commity ktoré sú ohľadom projektu.

Organizačné 2

- Overte si, že viete spraviť commit.
  Riešte problémy včas.

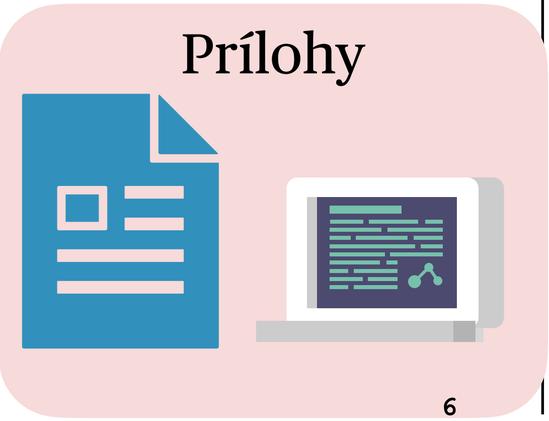
1. Zadanie

Odovzdáva sa vo fromáte PDF

- Mailová etiketa
  Predmet: **Vždy ho vyplňte**
  [OOP], OOP -, [CvUt14 OOP]
- Obsah mailu:
- Uveďte z ktorého ste cvičenia (úvod, záver)
- Github problém: uveďte svoj github handle
- Odstavce a odsadenie textu

- Predmet
  [OOP]Problém s odovzdaním úlohy task 7
- Obsah
  Dobrý deň,
  v piatok (27.02) som chcel odovzdať úlohu 7 do testovača ale vypísalo mi to nasledjúcu chybu (viď príloha). Rád by som sa spýtal, ako mám postupovať. Problém som riešil s cvičiacim, odkázal ma na Vás. Zip súbor prikladám ako prílohu.
  Ďakujem za Váš čas,
  Oliver Udvardi
  Utorok 14:00 cvičiaci: Revilo Idravdu

- Predmet
  [OOP]Problém s odovzdaním úlohy task 7
- Obsah
  Dobrý deň,
  v piatok (27.02) som chcel odovzdať úlohu 7 do testovača ale vypísalo mi to nasledjúcu chybu (viď príloha). Rád by som sa spýtal, ako mám postupovať. Problém som riešil s cvičiacim, odkázal ma na Vás. Zip súbor prikladám ako prílohu.
  Ďakujem za Váš čas,
  Oliver Udvardi
  Utorok 14:00 cvičiaci: Revilo Idravdu



Previously.
Dedenie. Konštruktor.
Overloading. Overriding.
Polymorfizmus. Abstrakcia.

Interface.

- Interface - úroveň abstrakcie
  Popisuje schopnosti (kontrakt), ktoré objekt poskytuje
- Napríklad Auto implementuje interface Elektrický pohon
- Elektrický pohon špecifikuje v rámci tela interface, ktoré metódy musia objekty implementovať
- Auto potom musí obsahovať tieto metódy aj s vnútornou implementáciou.

Demo

Akademická samospráva

Vzťahy medzi triedami.

Vzťahy v reálnom svete

Dedenie

Deti zdedia črty, správanie po rodičoch

Vlastní (je súčasťou)

Srdce je súčasť človeka

Pozná (komunikuje)

Človek používa mobil a mobil používa človeka

Má (jednosmerne)

Tím má hráčov

Vzťahy OOP

Dedenie

Deti zdedia črty, správanie po rodičoch

Vlastní (je súčasťou)

Srdce je súčasť človeka

Pozná (komunikuje)

Človek používa mobil a mobil používa človeka

Má (jednosmerne)

Tím má hráčov

Vzťahy OOP

Dedenie

class Child extends Parent

Vlastní (je súčasťou)

Srdce je súčasť človeka

Pozná (komunikuje)

Človek používa mobil a mobil používa človeka

Má (jednosmerne)

Tím má hráčov

Vzťahy OOP

Dedenie

class Child extends Parent

Kompozícia

class Human{…

Heart heart = **new Heart()**

…}

Pozná (komunikuje)

Človek používa mobil a mobil používa človeka

Má (jednosmerne)

Tím má hráčov

Vzťahy OOP

Dedenie

class Child extends Parent

Kompozícia

class Human{…
Heart heart = **new Heart()**

…}

Asociácia

class Human {Phone p;}
class Phone {Human h;}

Má (jednosmerne)

Tím má hráčov

Vzťahy OOP

Dedenie

class Child extends Parent

Kompozícia

class Human{…
Heart heart = **new Heart()**

…}

Asociácia

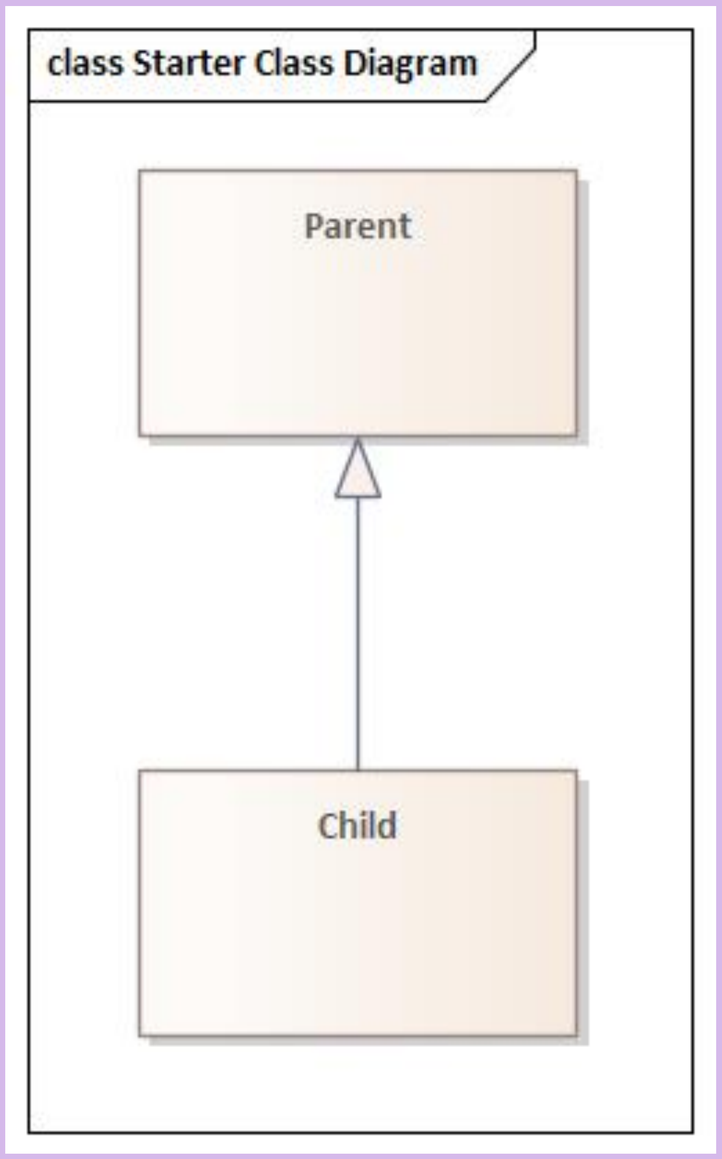
class Human {Phone p;}
class Phone {Human h;}

Agregácia

class Team{
ArrayList<Players> p;
}

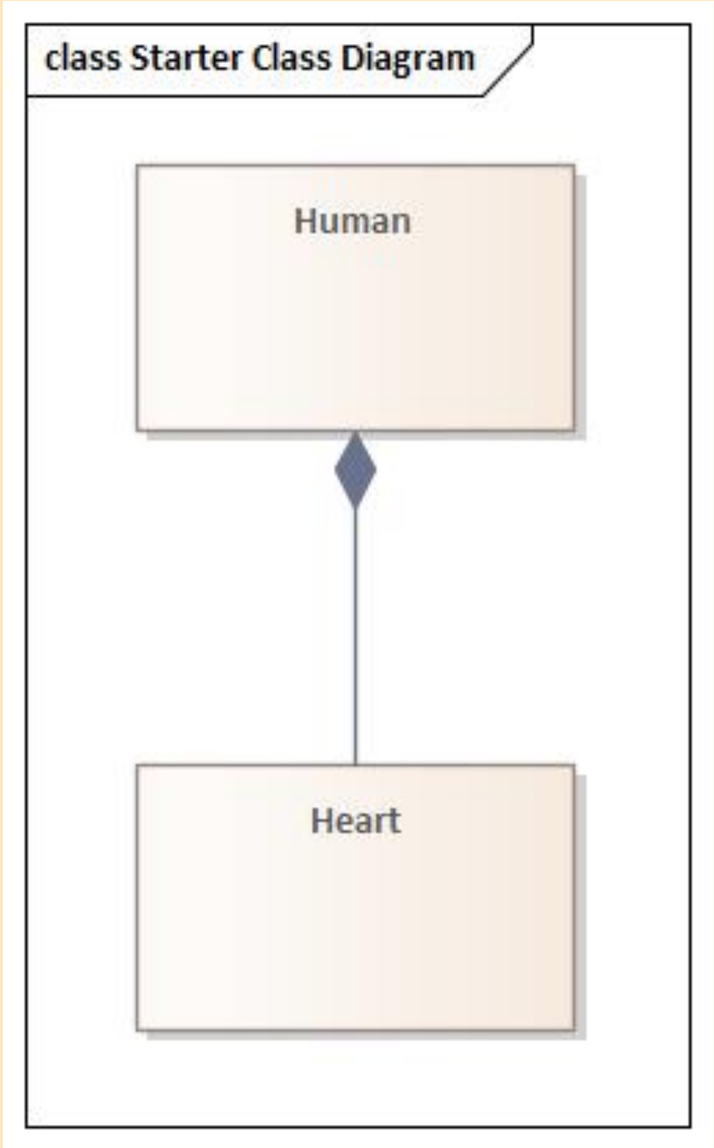
UML Dedenie

- Označuje sa prázdnym trojuholníkom
  Smer šípky:
  od Špecifický ––> k Všeobecný



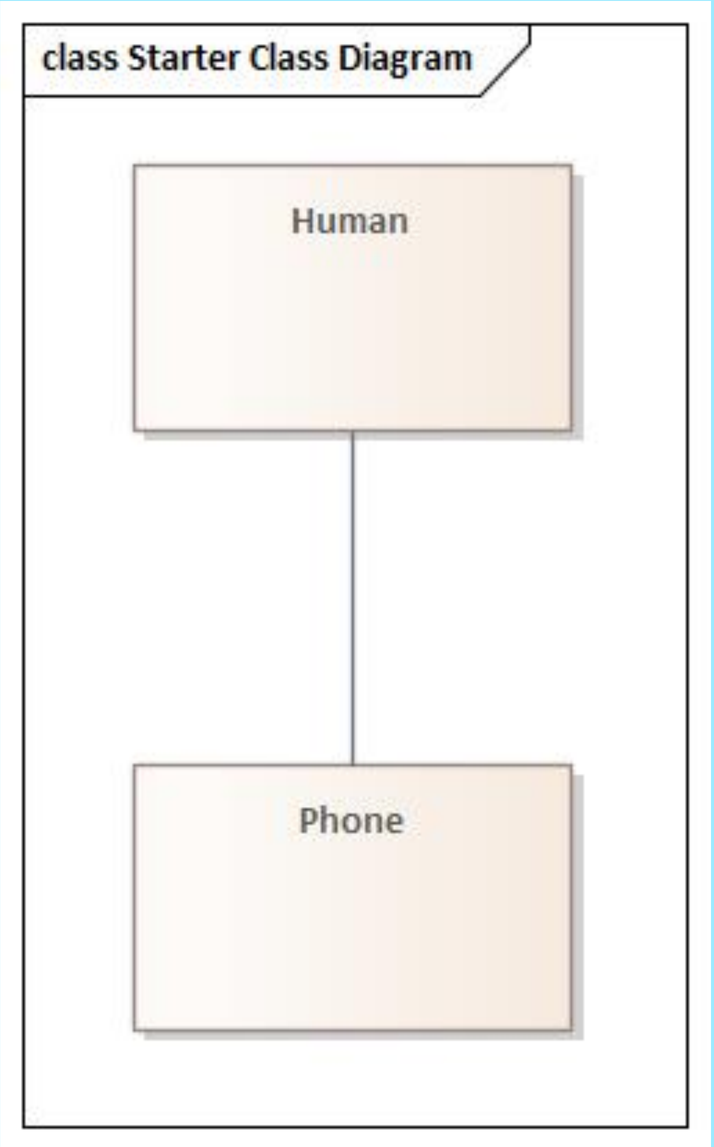
UML Kompozícia

- Označuje sa vyplneným diamantom
- Navigovateľnosť
  Celok je *Human*.



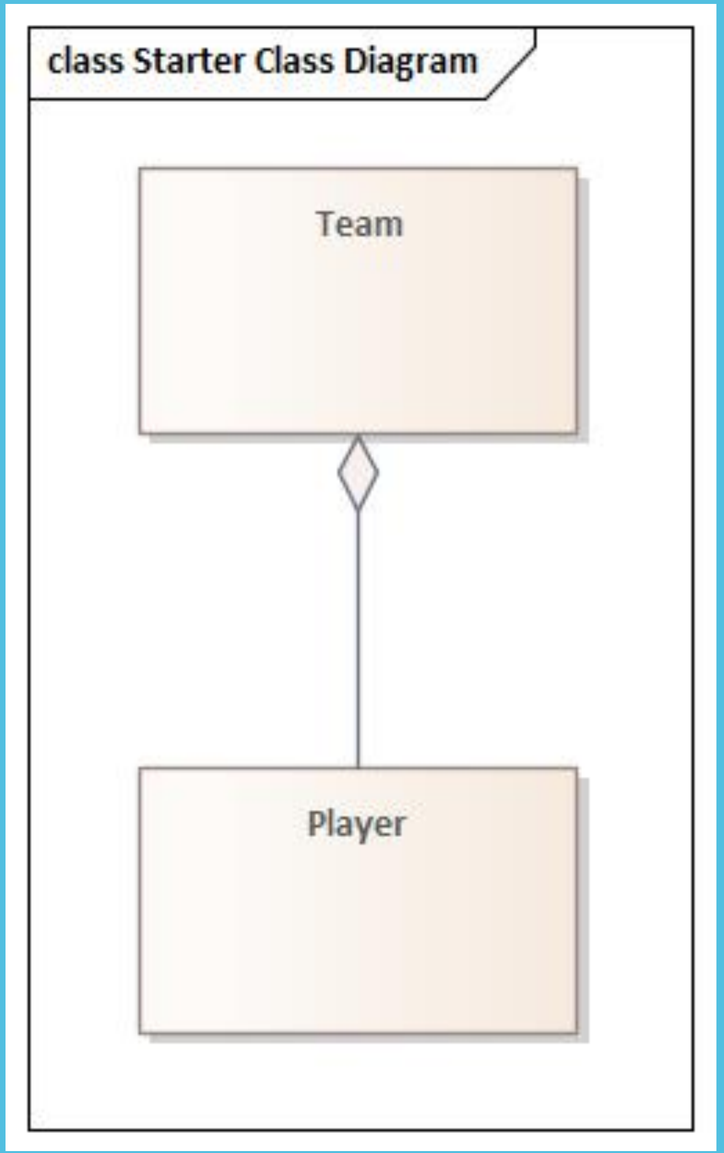
UML Asociácia

- Označuje sa plnou čiarou Vyjadruje všeobecný vzťah medzi triedami
- Navigovateľnosť (šípka) určuje, ktorá trieda má referenciu na druhú
- Bez šípky = navigovateľnosť nie je špecifikovaná



UML Agregácia

- Označuje sa prázdnym diamantom
- Navigovateľnosť
  Kto koho pozná. Celok je *Team*



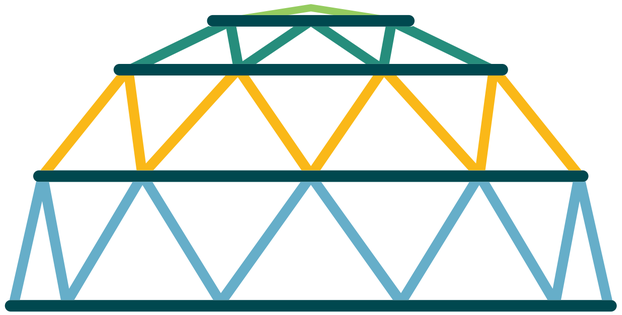
Návrhové vzory.

- Vaše problémy už niekto zažil pred vami.
- A vyriešil ich.
- Vzor
- Zvykne sa opakovať v rôzných kontextoch.
- Byť vzorom - best practise

Klasifikácia

Creational Patterns

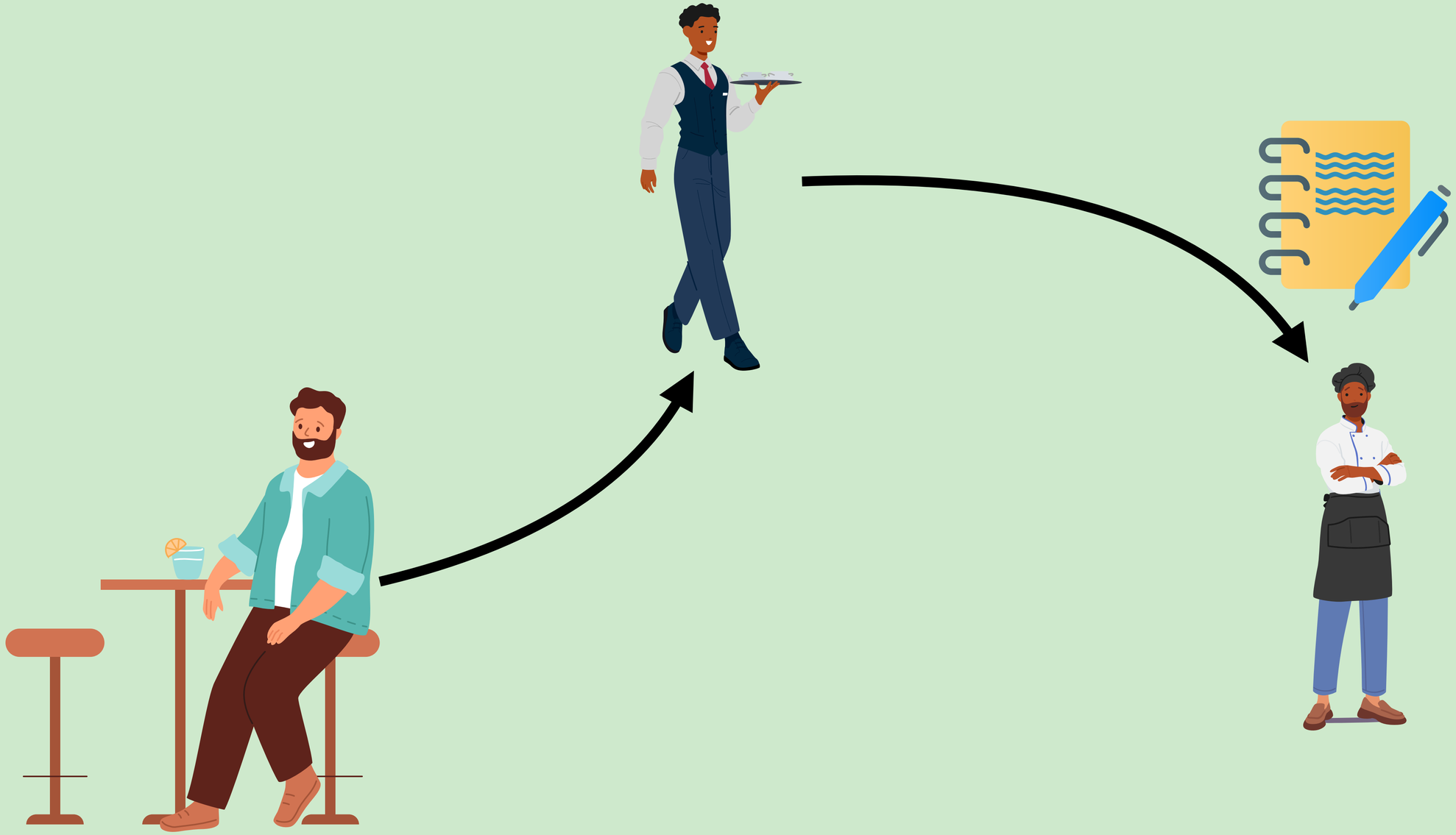
Behavioral Patterns

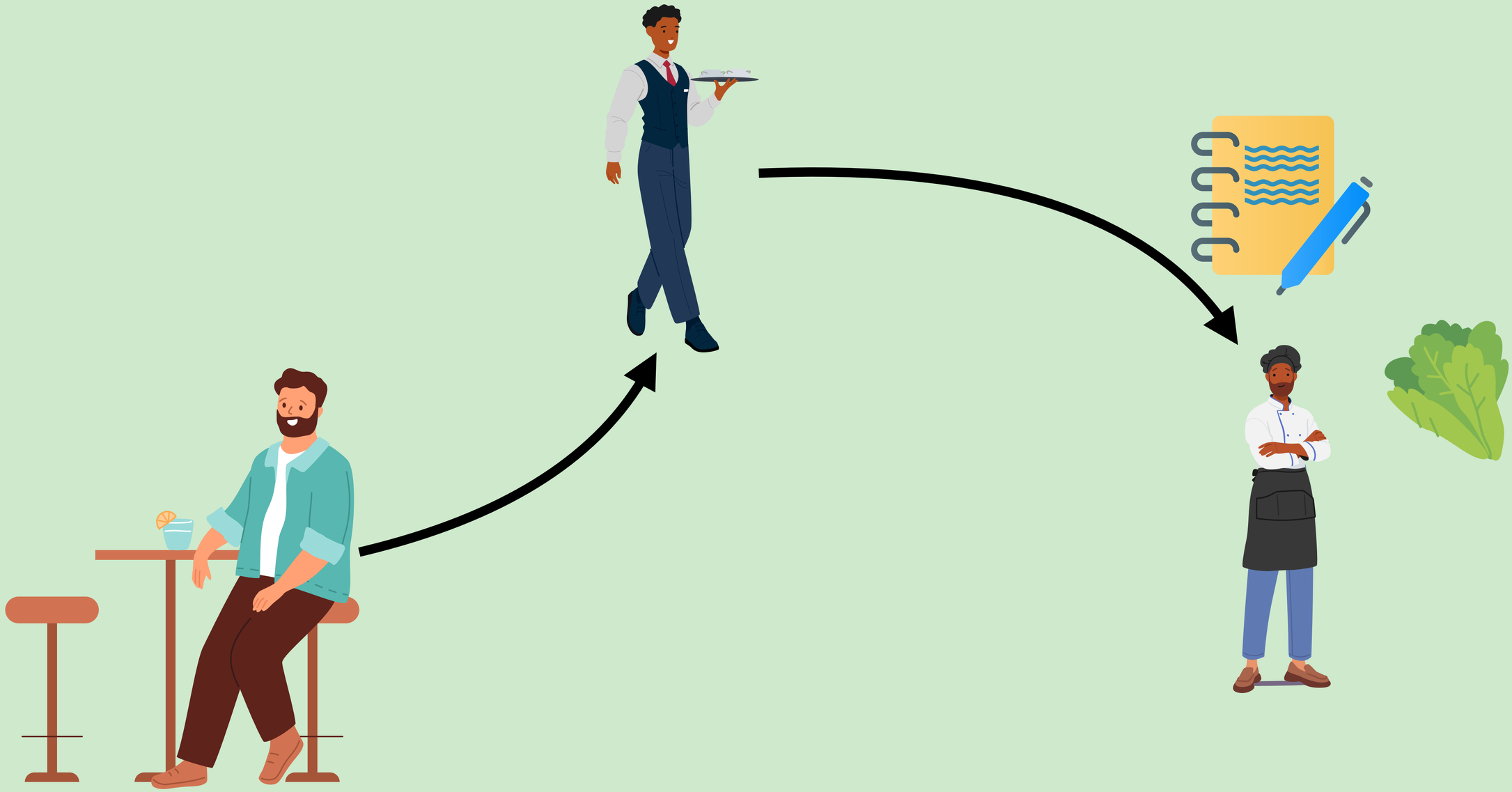


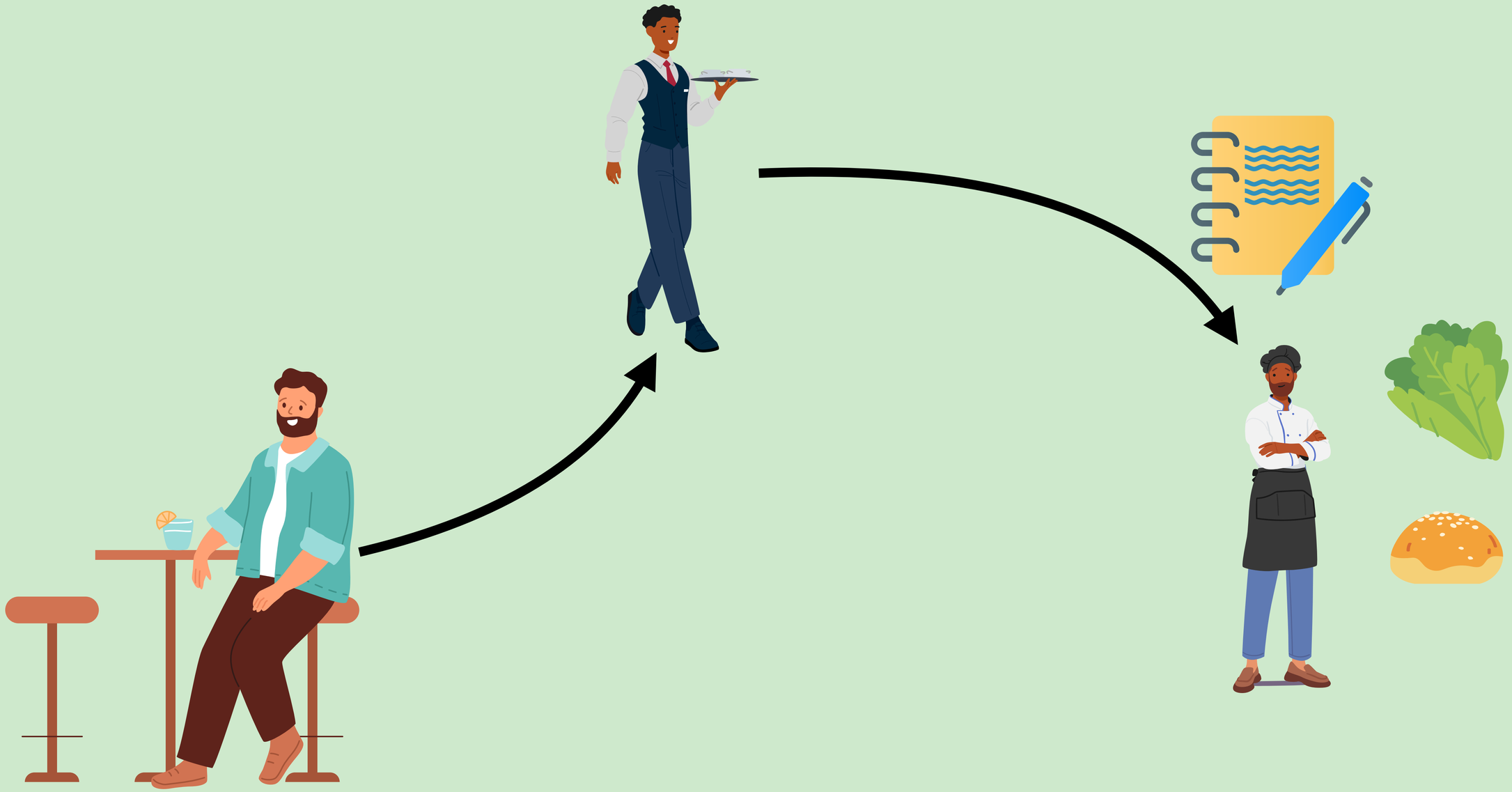
Structural Patterns

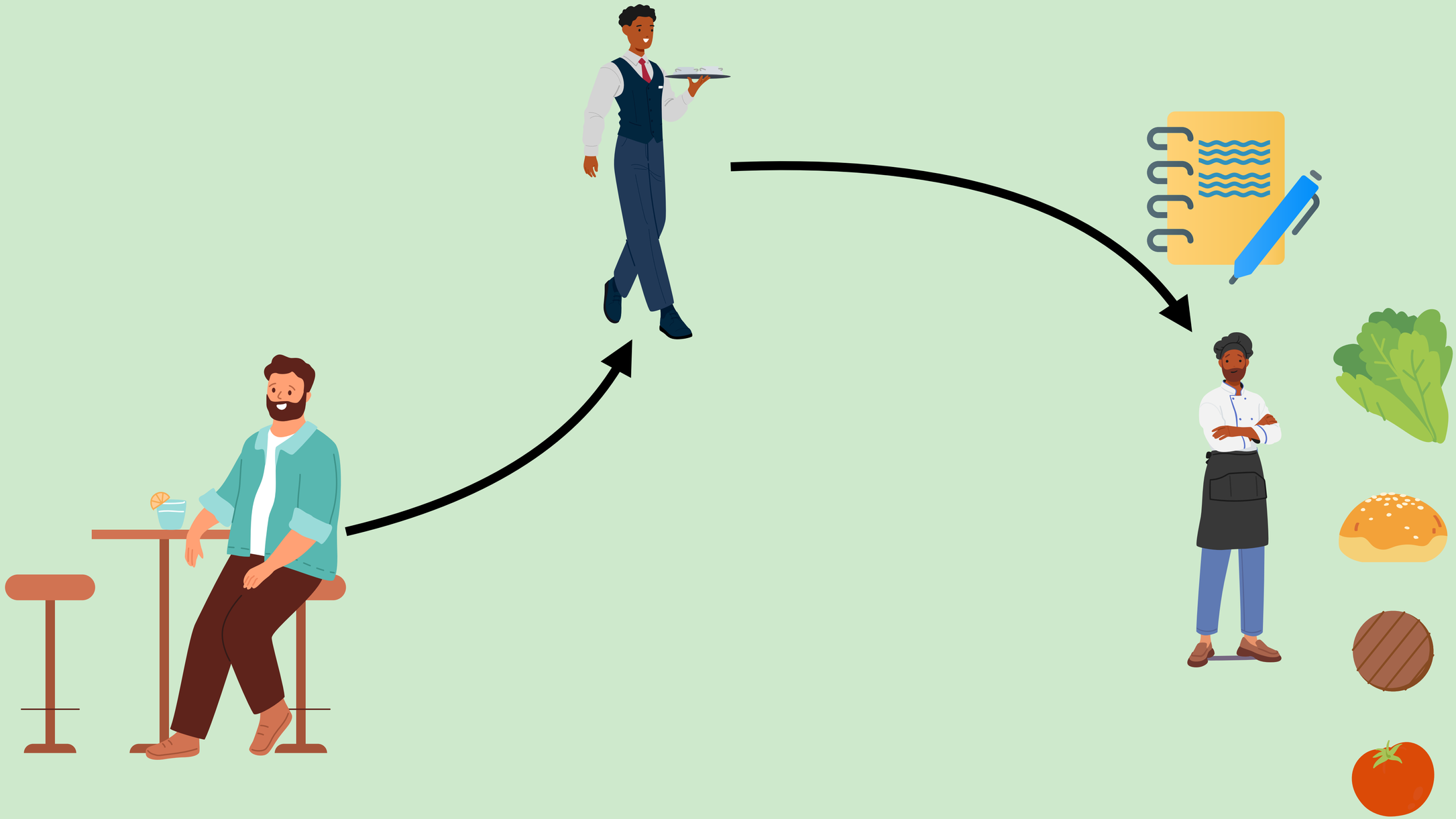
- Model View Controller
- Model View Presenter

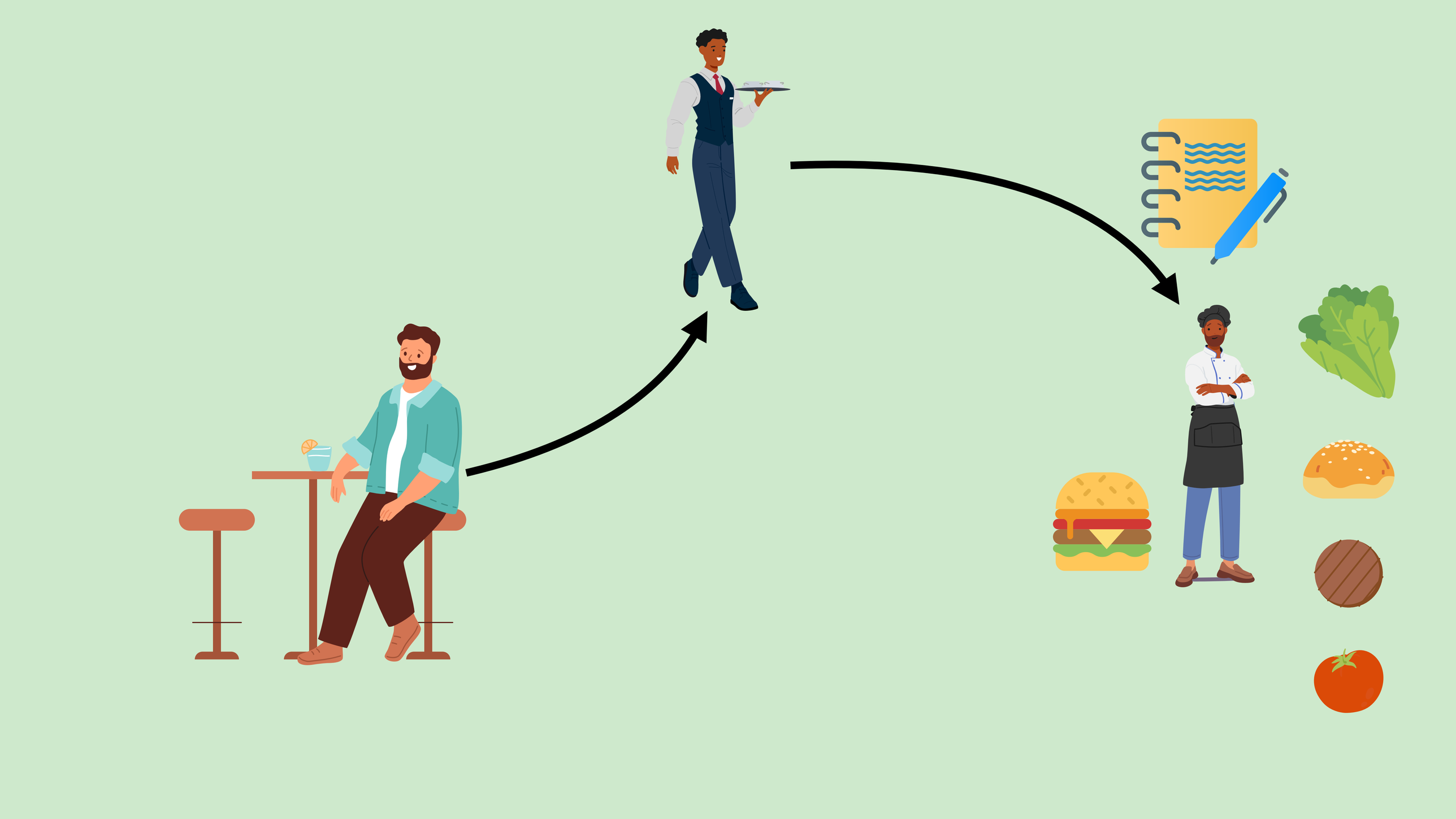
Model View Controller.

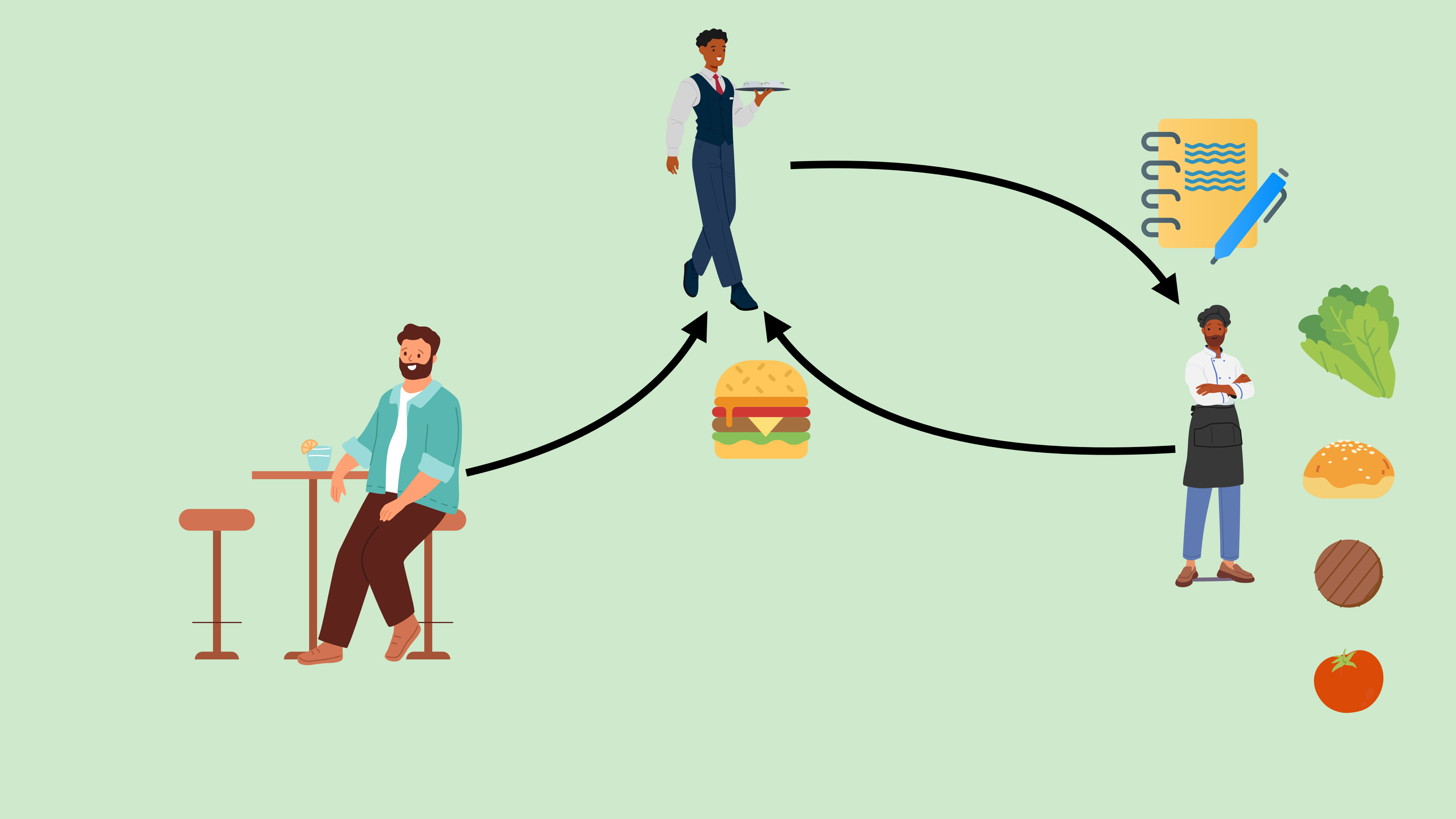


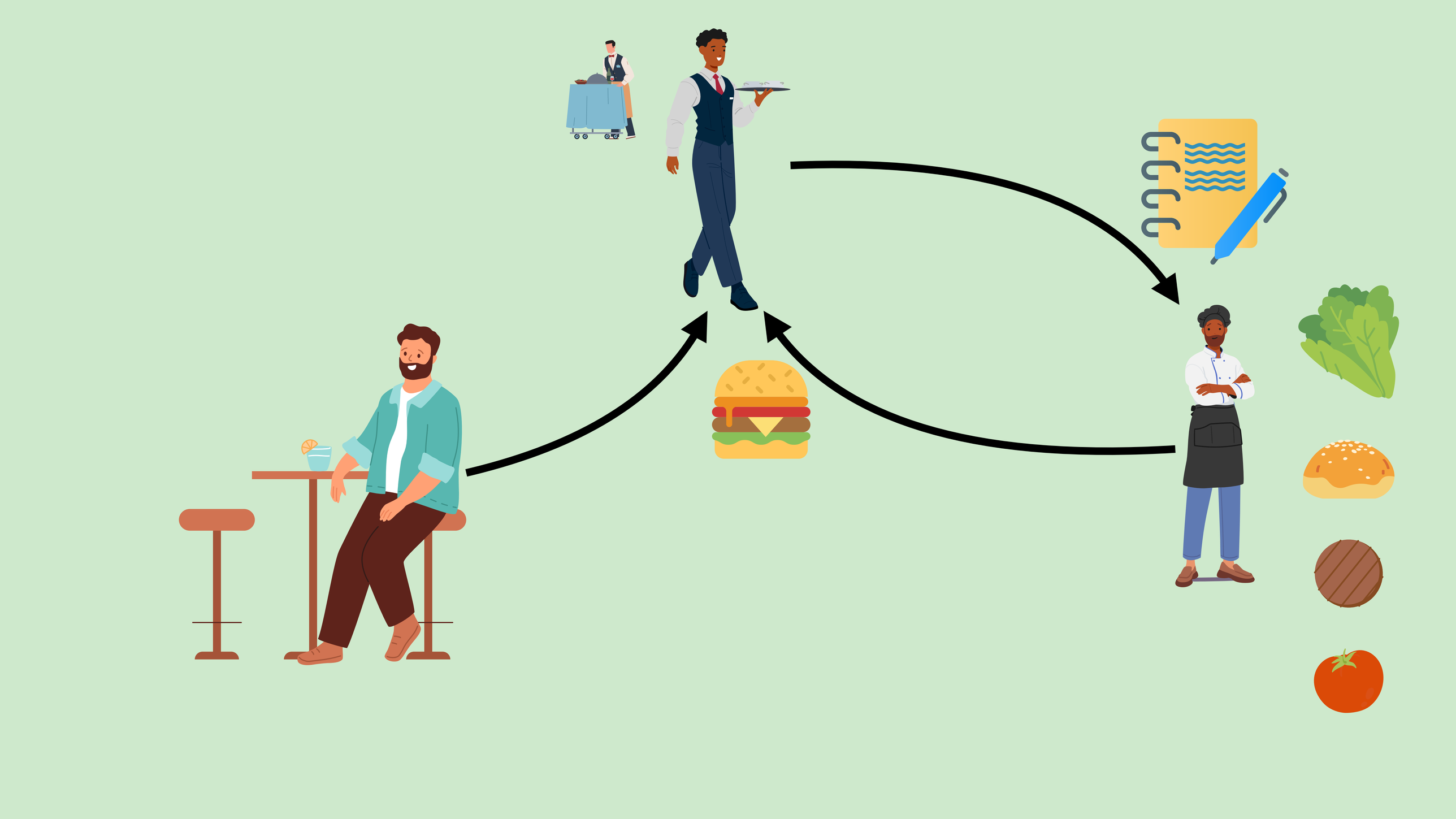


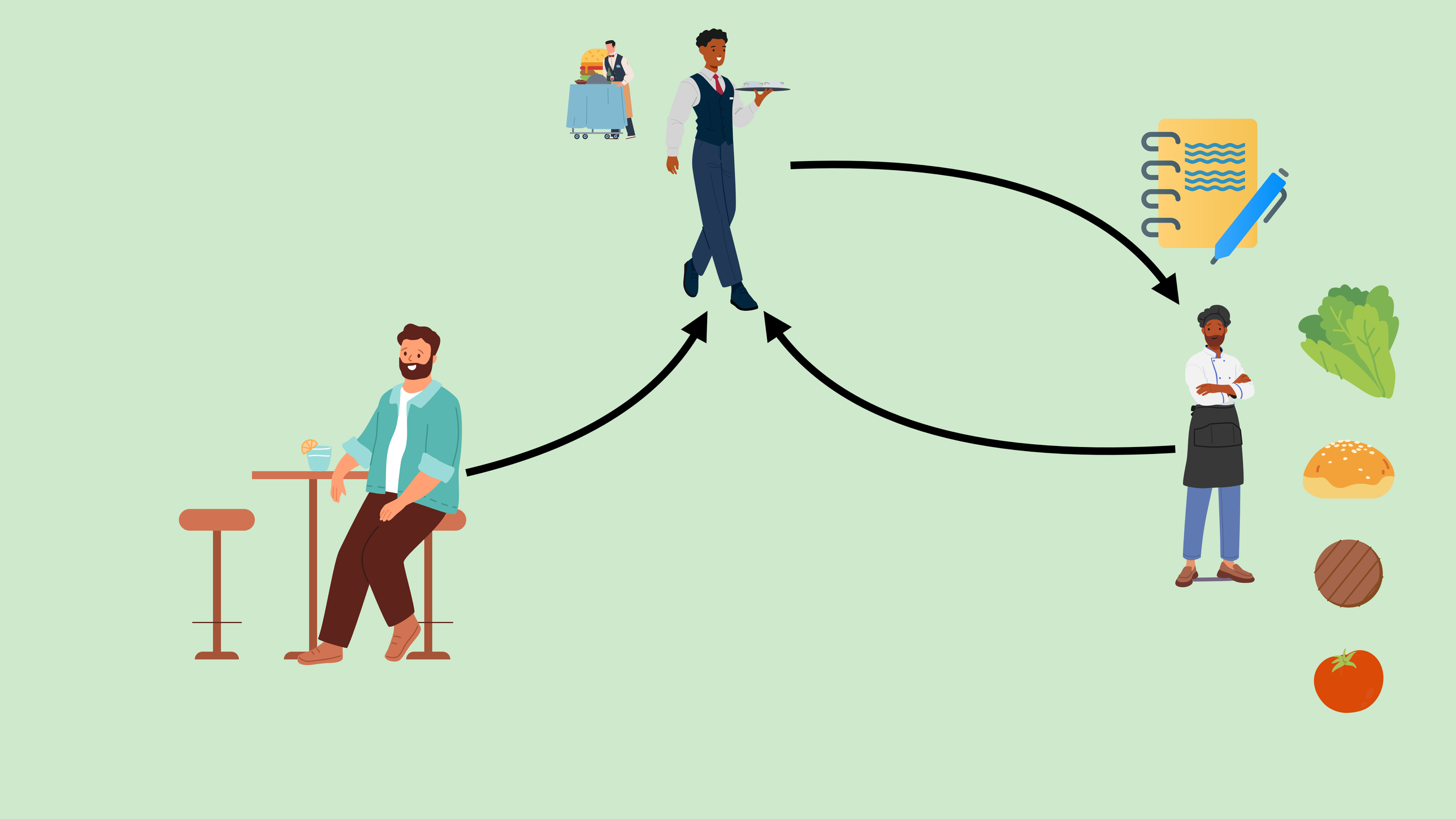


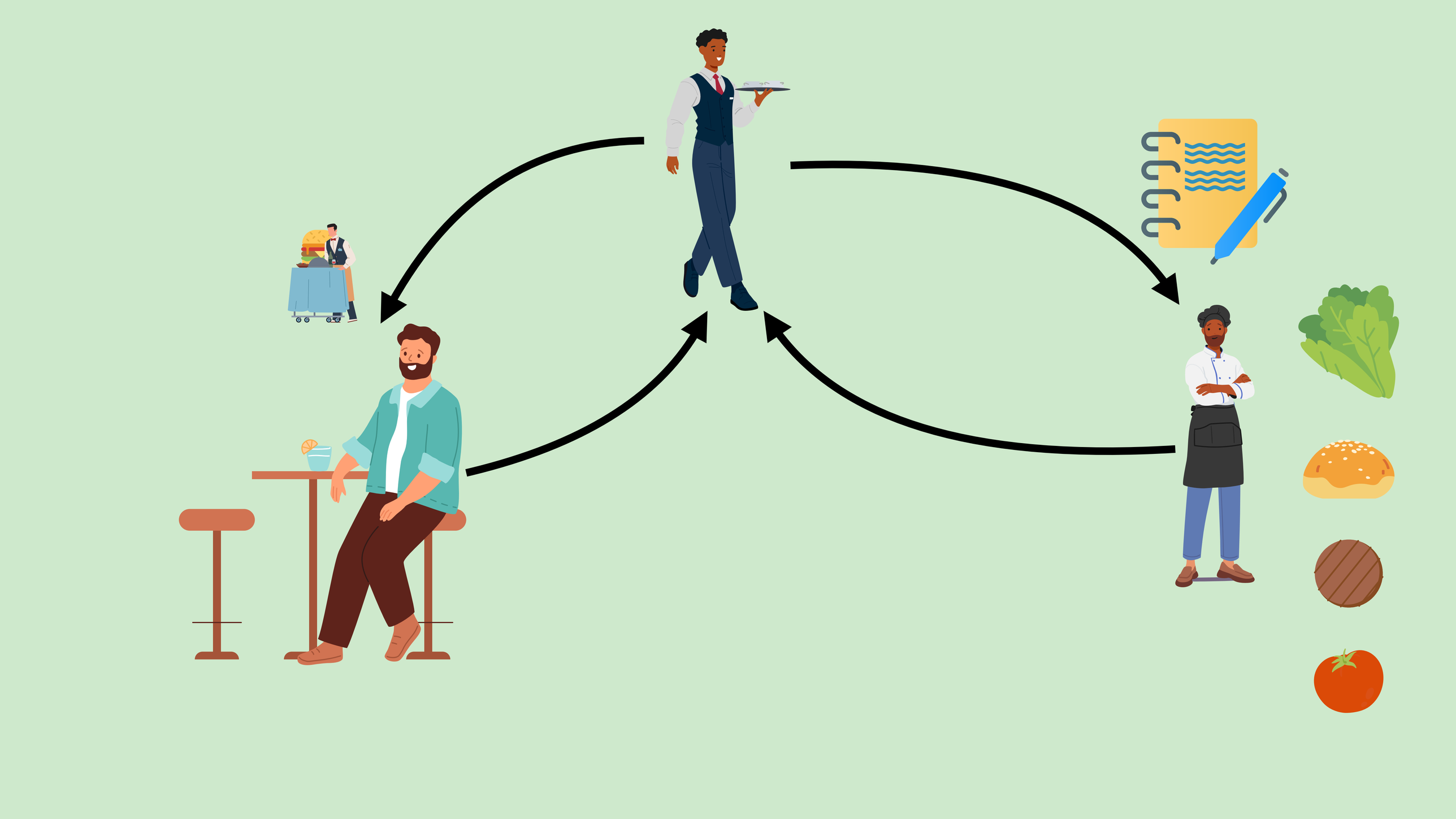


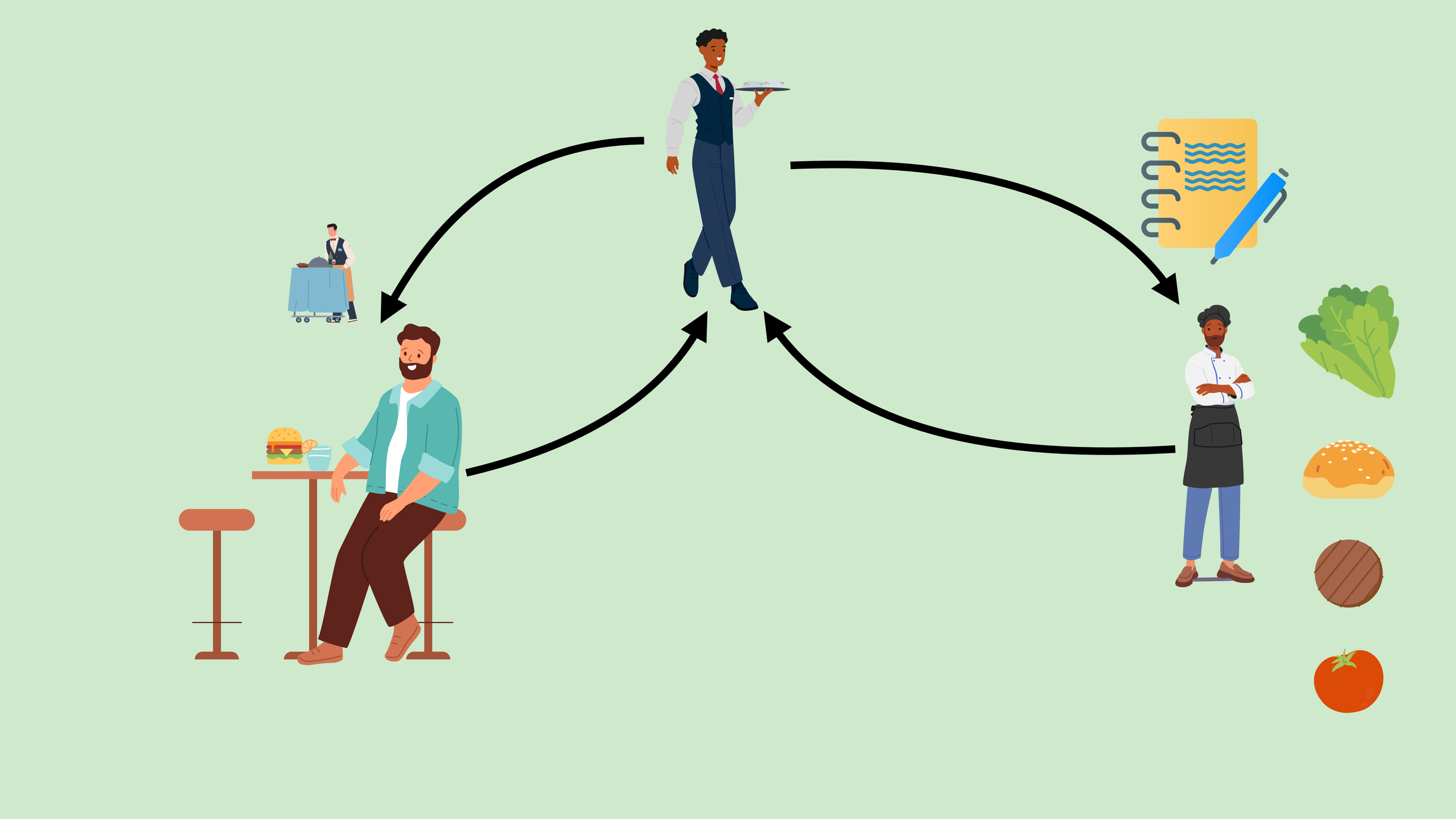


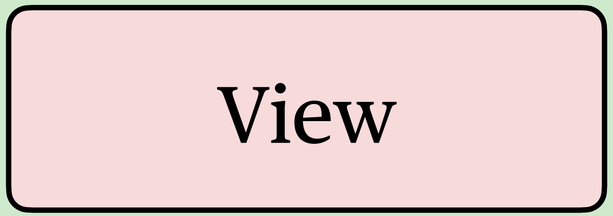











View

View

View

View

View

Model

Stavy objektov

Volanie metód na zmenu

Controller

View

Model

Stavy objektov

Volanie metód na zmenu

Controller

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

Controller

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

Controller

1

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

Controller

1

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Controller

1

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Controller

1

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Controller

3

Posuň postavičku +1

1

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Controller

3

Posuň postavičku +1

1

Šípka vpravo

View

Model

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Controller

3

Posuň postavičku +1

1

Šípka vpravo

Zmena stavu objektu

View

Model

4

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Controller

3

Posuň postavičku +1

1

Šípka vpravo

Zmena stavu objektu

View

Model

4

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Controller

3

Posuň postavičku +1

1

Šípka vpravo

5

Vráť nový stav

Zmena stavu objektu

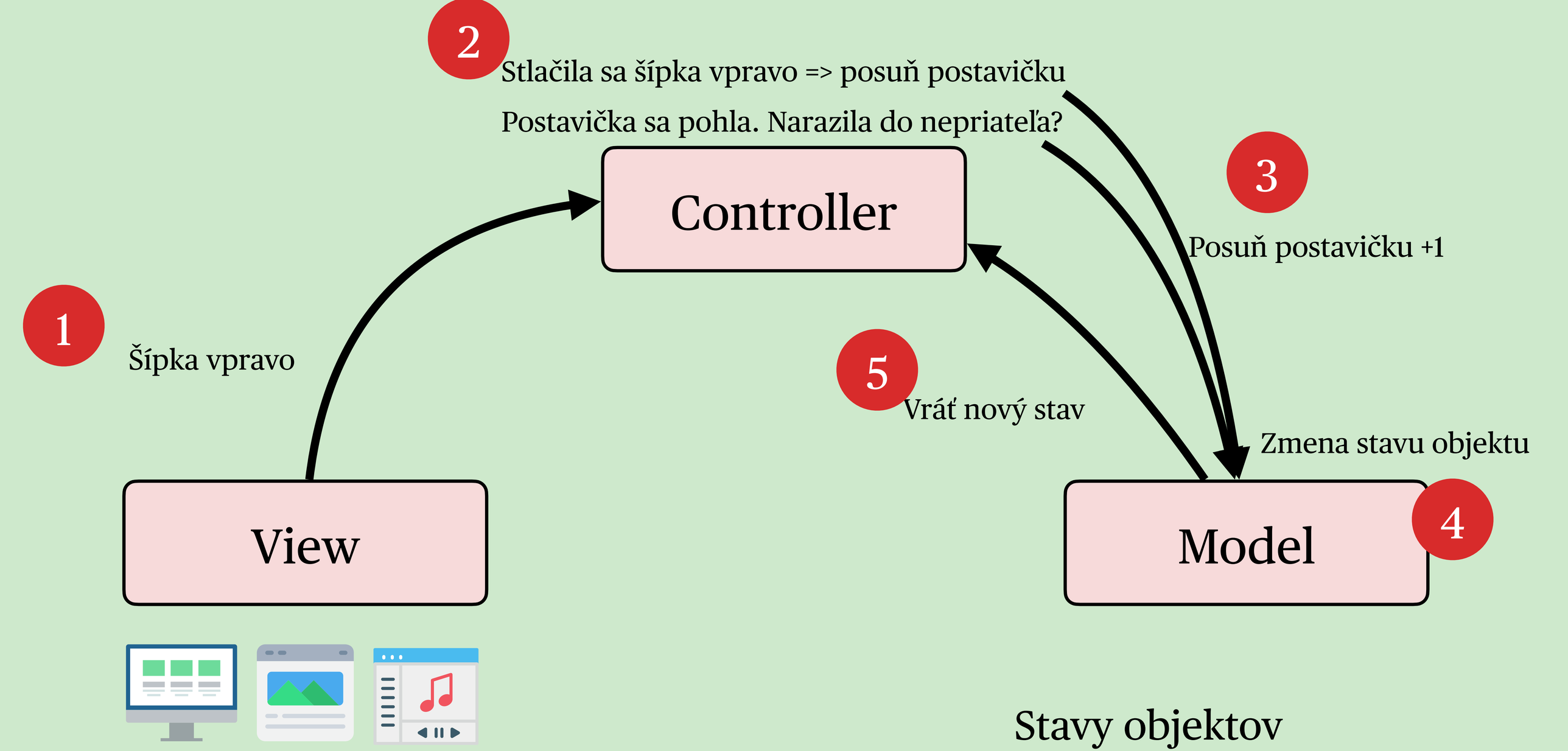
View

Model

4

Stavy objektov

Volanie metód na zmenu



Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Postavička sa pohla. Narazila do nepriateľa?

Controller

3

Posuň postavičku +1

1

Šípka vpravo

5

Vráť nový stav

Vráť aktuálny stav

Zmena stavu objektu

View

Model

4

Stavy objektov

Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Postavička sa pohla. Narazila do nepriateľa?

Controller

3

Posuň postavičku +1

1

Šípka vpravo

5

Vráť nový stav

Vráť aktuálny stav

Zmena stavu objektu

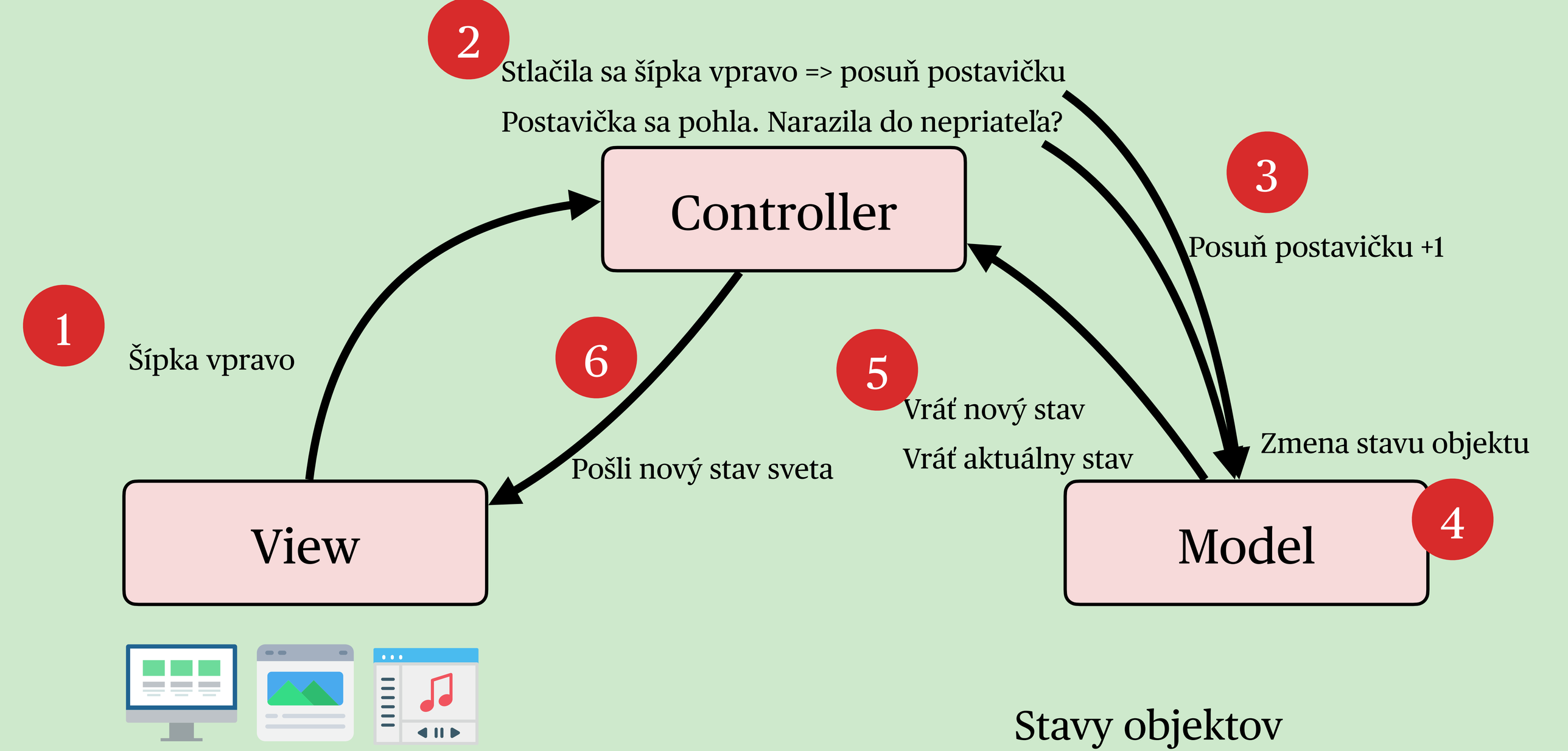
View

Model

4

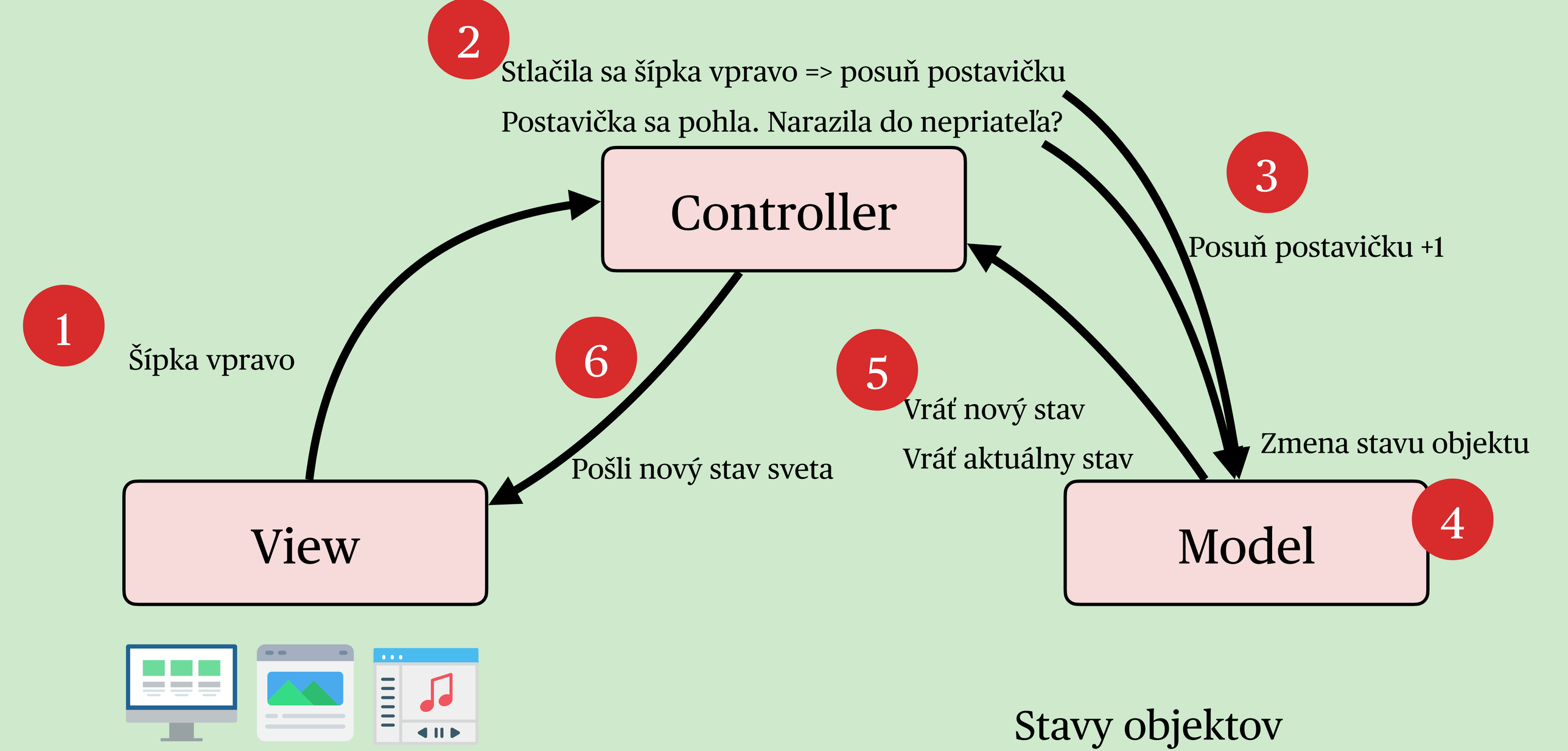
Stavy objektov

Volanie metód na zmenu



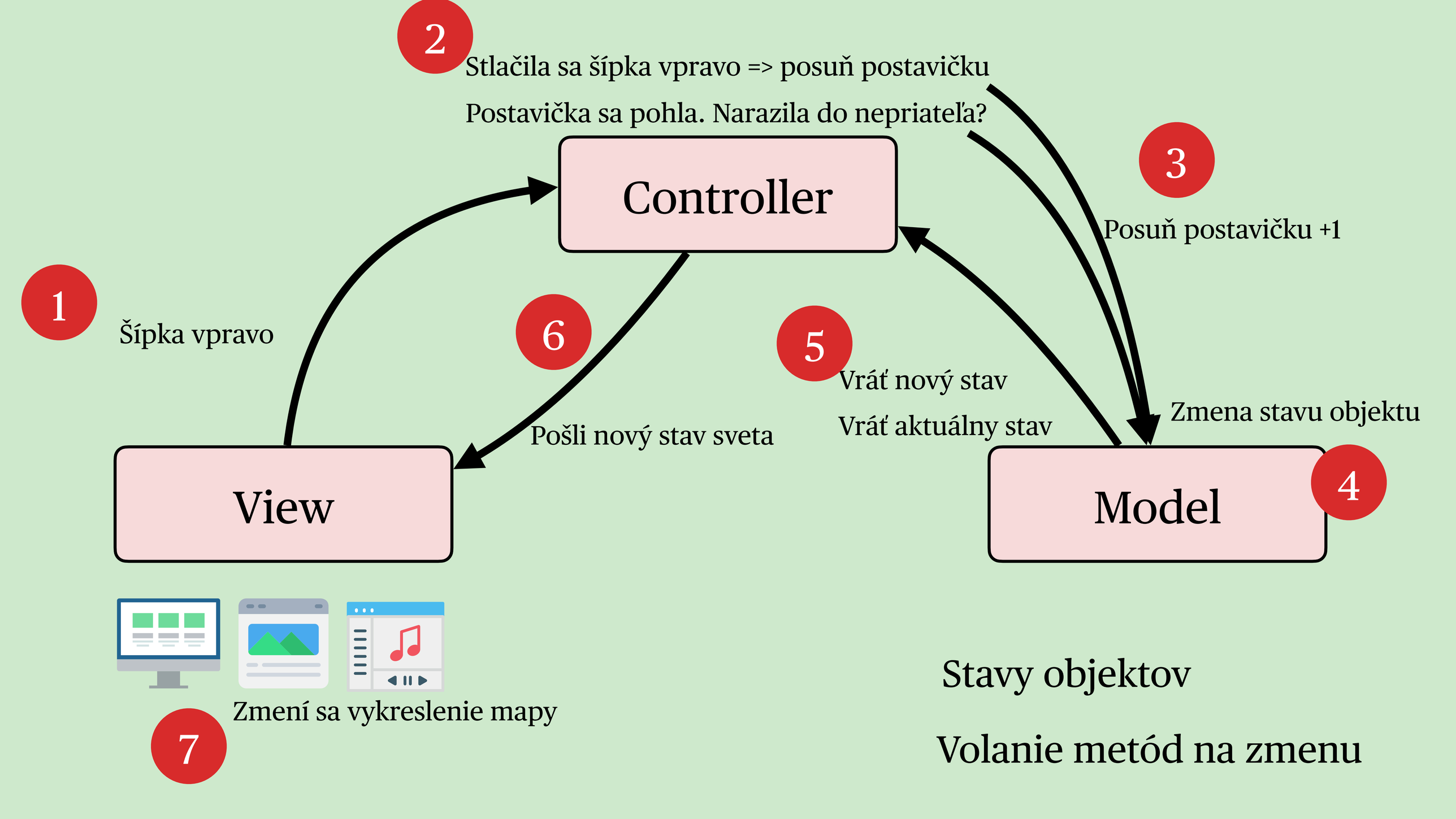
Stavy objektov

Volanie metód na zmenu



Stavy objektov

Volanie metód na zmenu


2
Stlačila sa šípka vpravo => posuň postavičku
Postavička sa pohla. Narazila do nepriateľa?
Controller
3
Posuň postavičku +1
1
Šípka vpravo
6
5
Vráť nový stav
Vráť aktuálny stav
Zmena stavu objektu
Pošli nový stav sveta
View
Model
4
Zmení sa vykreslenie mapy
7
Stavy objektov
Volanie metód na zmenu

2

Stlačila sa šípka vpravo => posuň postavičku

Postavička sa pohla. Narazila do nepriateľa?

Controller

3

Posuň postavičku +1

1

Klik na End

Šípka vpravo

6

5

Vráť nový stav

Vráť aktuálny stav

Zmena stavu objektu

Pošli nový stav sveta

View

Model

4

Zmení sa vykreslenie mapy

7

Stavy objektov

Volanie metód na zmenu

1

Klik na End

Šípka vpravo

2

Stlačila sa šípka vpravo => posuň postavičku

Postavička sa pohla. Narazila do nepriateľa?

Controller

3

Posuň postavičku +1

Zmena stavu objektu

Model

4

Zmenili sa stavy objektov

Môže byť aktualizácia priamo z modelu. Stále je to MVC.

5

View

Zmení sa vykreslenie mapy

6

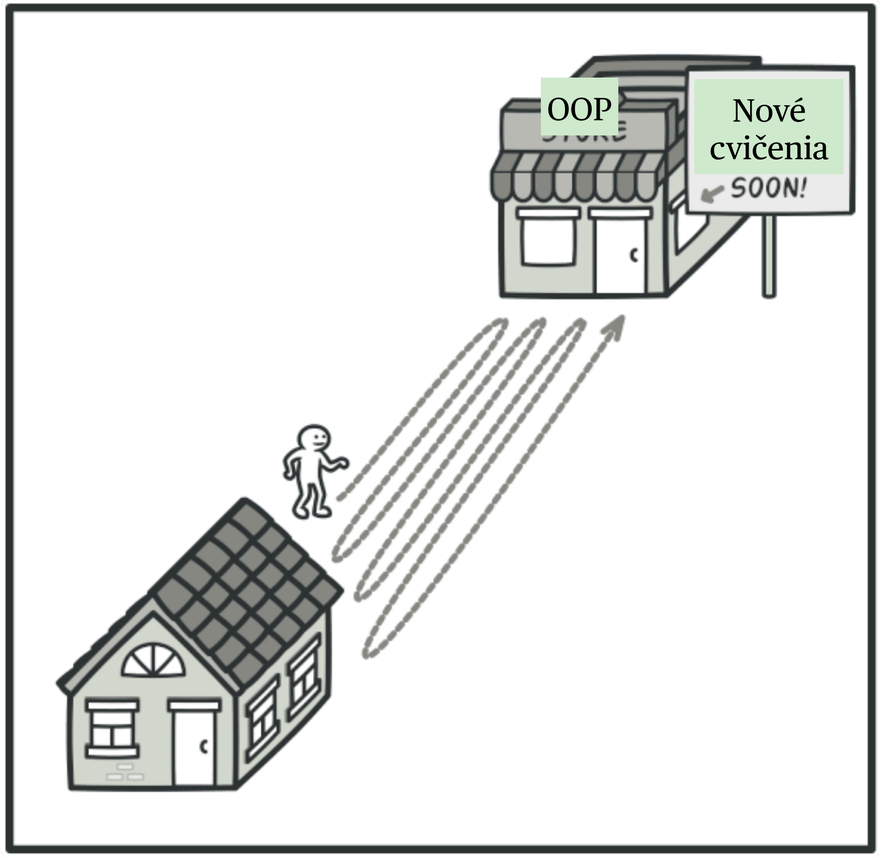
Stavy objektov

Volanie metód na zmenu

Observer.

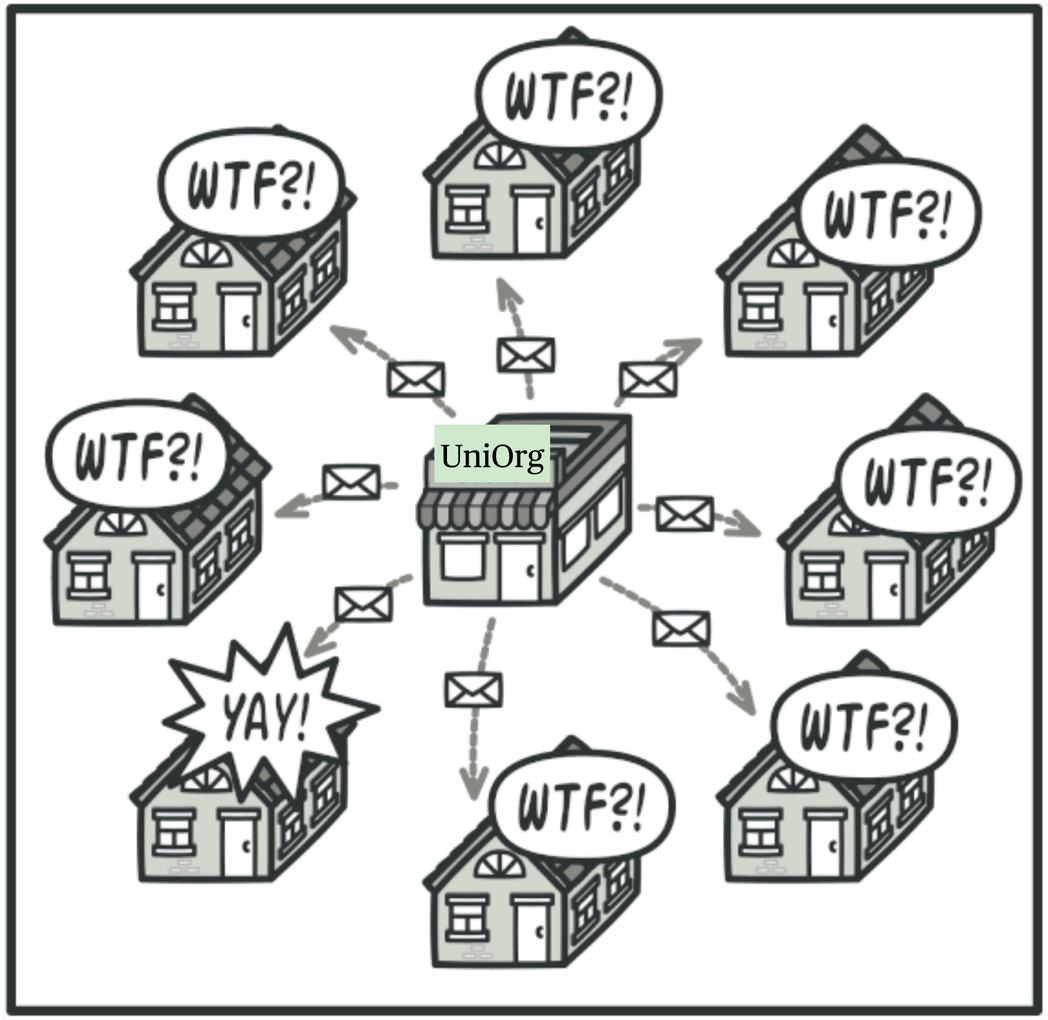
Problém

- Na stránke oop.fiit.stuba.sk sa zverejňujú informácie ale je to nepravidelne. Pravidelne to chodíte kontrolovať, ale je to únavné.



SOURCE: REFACTORING.GURU

Problém 2 · Univerzitná organizácia chce študentov informovať o novinkách a (pre ňu) veľmi dôležitých informáciach

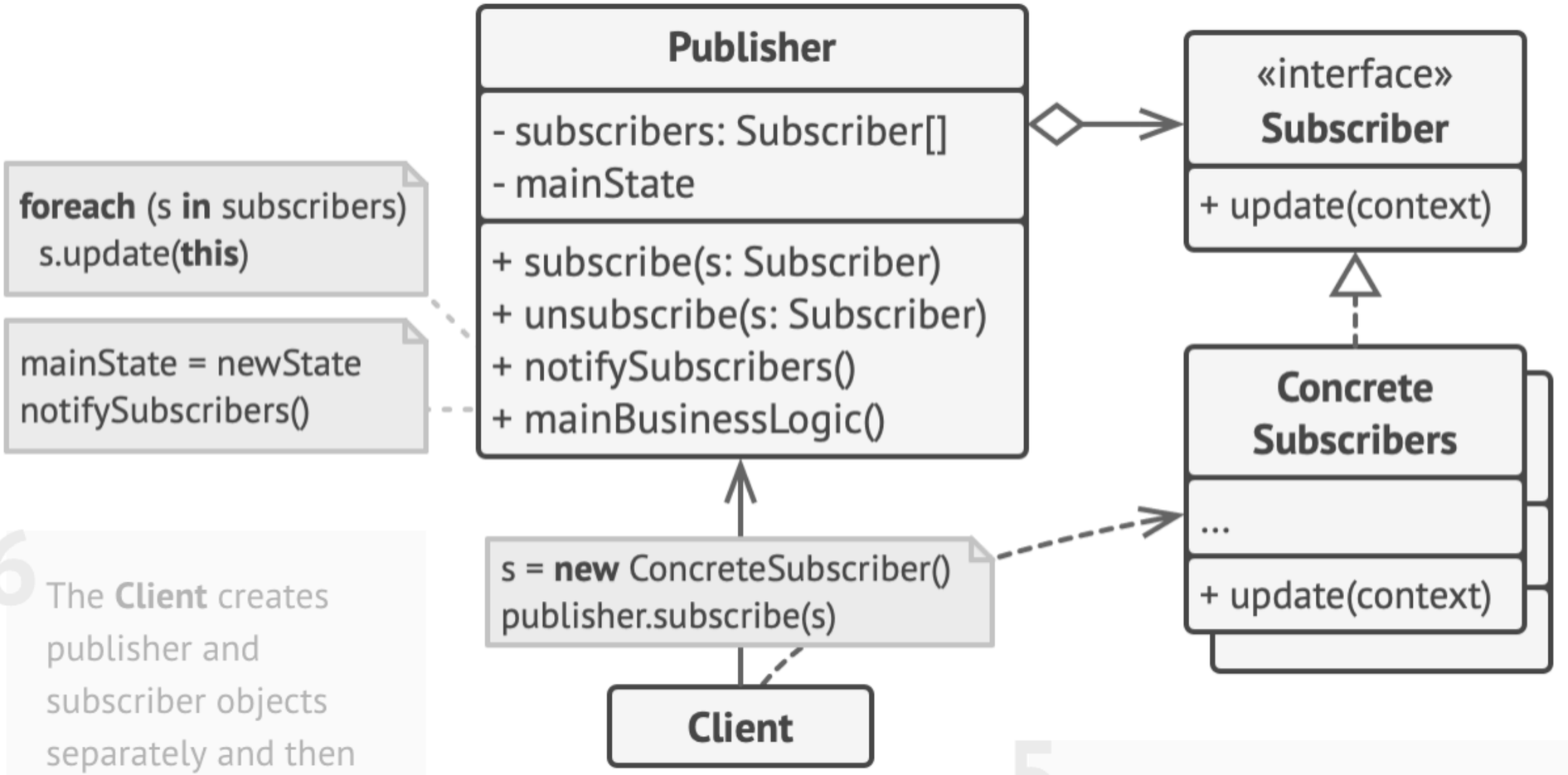


SOURCE: REFACTORING.GURU

Riešenie

- Vytvoríte Triedu, ktorá bude pri zmene stavu alebo keď nastane nejaký *trigger* informovať odoberateľov, ktorí sa sami k takémuto odberu prihlásili. (YouTube subscribers)

- Cez interface vynútite, aby každý odoberateľ implementoval metódu, ktorou ho viete notifikovať (napr. `update()`)



SOURCE: REFACTORING.GURU

Strategy.

Problém

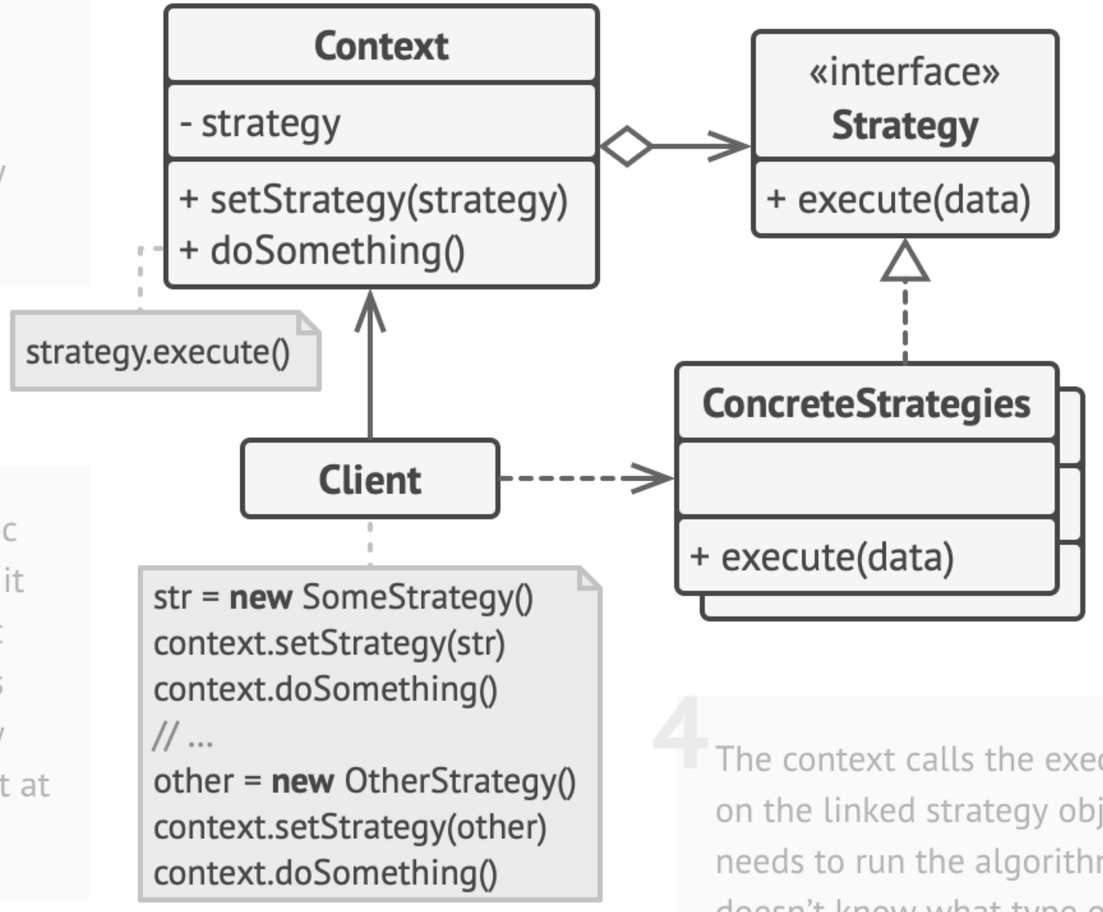
- Navigácia - najlepšia cesta - čo to znamená?
- Nakupovanie - najlepší deal - čo to znamená?

- Navigácia - nejrýchlejšia cesta, udržateľná cesta, bez dialních
- Nakupovanie - materiály, značka, cena, kupóny

Riešenie

- Vytvoríte si Interface, ktorom si vynútite vykonanie (`execute()`) stratégie, ktorú si každá stratégia implementuje samostatne.
- Context (Navigácia, nakupovanie) - obsahuje atribút stratégie.
- Vytvoríte metódu, ktorou môžete podľa potreby meniť referenciu na inú stratégiu

1 The **Context** maintains a reference to one of the concrete strategies and communicates with this object only via the strategy interface.



2 The **Strategy** interface is common to all concrete strategies. It declares a method the context uses to execute a strategy.

3 **Concrete Strategies** implement different variations of an algorithm the context uses.

4 The context calls the execution method on the linked strategy object each time it needs to run the algorithm. The context doesn't know what type of strategy it works with or how the algorithm is executed.

5 The **Client** creates a specific strategy object and passes it to the context. The context exposes a setter which lets clients replace the strategy associated with the context at runtime.

SOURCE: REFACTORING.GURU

Singleton.

Problém

- Potrebujete zaručiť, že nejaká Trieda (hlavný controller aplikácie, databáza, Hra,...) **bude mať len jednu inštanciu**

- Potrebujete k nejakej triede pritúpiť ako ku globálnej premennej. Z akéhokoľvek miesta v programe

Riešenie

- Singleton!

1. Zabránite vytvorenie inštancie triedy externým objektom. **Konštruktor spravíte privátny.**
2. Vytoríte verejnú, statickú metódu na získanie inštancie samého seba. V tejto metóde zavoláte privátny konštruktor ak ešte inštancia nebola vytvorená. Inak vrátite už vytvorenú inštanciu.

```
public class UniverzitnySenat {

    private static UniverzitnySenat instance;

    private UniverzitnySenat() {}

    public static synchronized UniverzitnySenat getInstance() {
        if (instance == null) {
            instance = new UniverzitnySenat();
        }
        return instance;
    }
}
```

Otázky?

Quiz time!

OOP Kvíz 3