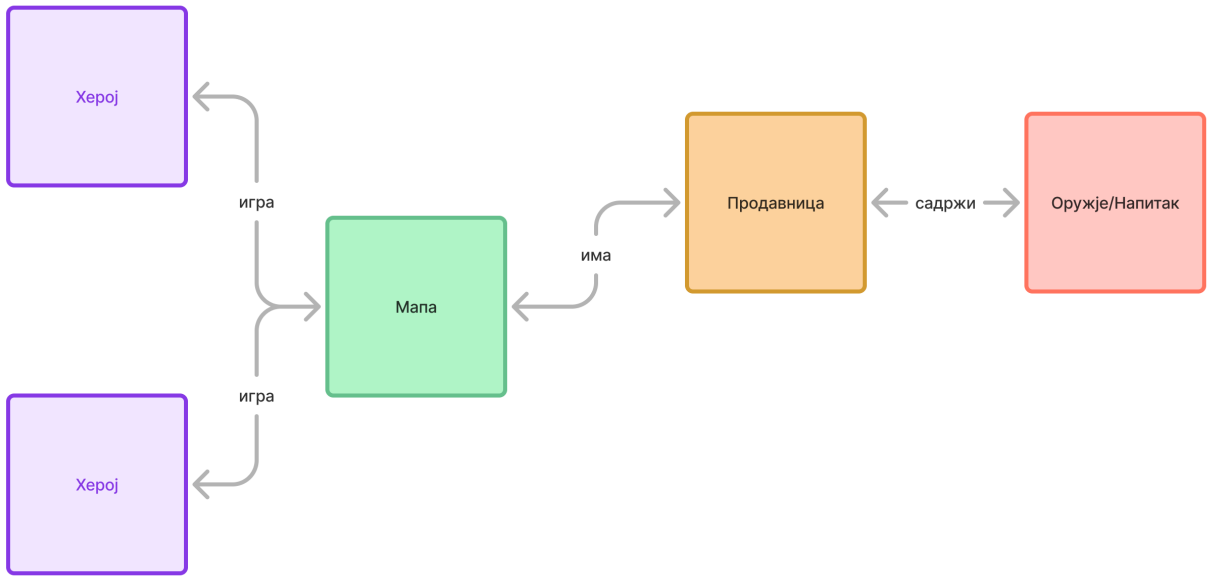
Пројектни задатак E03

Креирати, тестирати и симулирати систем онлајн ***MOBA*** игрице. Неопходно је направити **дизајн система**, креирати на основу слике E03.1 **архитектуру система**, извршити **имплементацију** и програмско решење адекватно **тестирати** употребом предефинисаних тестова.



Слика E03.1: Архитектура система

Игрица се састоји од **хероја** који је описан следећим атрибутима:

- **Јединствени назив хероја**
- **Број животних поена** (целобројна вредност)
- **Јачина напада у животним поенима** (целобројна вредност)
- **Тренутно стање златних новчића** (целобројна вредност)

Хероји се налазе на **мапи** за игру описаном следећим атрибутима:

- **Назив мапе** (јединствен)
- **Тип мапе** (зимска или летња верзија)
- **Максималан број играча** (целобројна вредност)
- **Назив плавог тима који тренутно игра на мапи**
- **Назив црвеног тима који тренутно игра на мапи**
- **Број помоћних ентитета доступних за освајање новчића**

Хероји могу да купују оружја и/или магичне напитке која им појачавају способности у борби из **продавнице** која је описана следећим атрибутима:

- **Јединствени идентификатор продавнице** (целобројна вредност)
- **Листа доступног оружја за куповину**
- **Листа доступних напитака за куповину**
- **Укупна вредност продатих артикала** (оружје + напици)

Оружје односно магичан напитак описан је следећим атрибутима:

- **Назив** (јединствено)
- **Цена за један комад**
- **Број појачаних поена за напад** (вредност између 15 и 40 поена)
- **Доступна количина за куповину** (целобројна вредност)

Оружје и напици могу се куповати валутом у игрици – **златни новчић**. Златни новчићи се могу освојити на један од 2 начина:

- **Елиминацијом противничког хероја** (награда је 300 златних новчића)
- **Елиминацијом помоћних ентитета** (награда је 20-90 златних новчића)

Путем конзолне апликације која приликом покретања захтева некакав вид **аутентификације**, омогућити унос **свих ентитета**. Унос ентитета је могуће реализовати и употребом методе за насумично генерисање објекта ентитета.

Омогућити симулацију покретање битке између тимова. Уноси се: назив **мапе** на ком ће се одвијати битка (сви параметри за одабрану мапу морају раније бити унети), затим **продавница** са оружјима и напицима који ће бити доступни у тој битци. Након уноса мапе, хероја и продавнице уноси се **број играча по тиму**, након чега се прво уносе називи играча за **плави тим**, а затим називи играча за **црвени тим** након чега почиње битка која траје између **10** и **45** секунди (трајање битке је насумично одређено).

У току битке неопходно је симулирати неколико елиминација противничког хероја и помоћних ентитета. Начин на који ће се одузимати животни поени хероја је произвољан, али је неопходно урачунати у то тренутну статистику хероја који напада. Када неки херој скупи **500** златних новчића извршити куповину из продавнице и херојy урачунати додатне поене за напад.

Након завршене битке приказује се опција за **приказ статистике битке**. Статистика се може **табеларно исписати на екрану** или сачувати у **текстуалну датотеку**. У случају да се одабере чување у текстуалну датотеку потребно је унети назив датотеке у којој ће се резултати сачувати.

Приказ статистике битке неопходно је урадити кроз ***Depedency Inversion*** принцип. Метода независно од начина прима **листу** хероја плавог тима и **листу** хероја црвеног тима и онда исписује на екран односно у текстуалну датотеку уписује: **информације о херојy** за оба тима, **назив мапе** на којој се битка десила, и укупну вредност продатих ставки из продавнице.

Напомене

Потребно је искористити макар један развојни образац приликом имплементације решења препорука је да то буде ***Singleton*** развојни обрасац.

Одговарајући начин приказа статистике битке **инјектује** се кроз конструктор класе сервиса за **статистику**, након чега се извршава одговарајући позив имлементације – ***Depedency Inversion*** принцип.

Неопходно је испоштовати ***SOLID*** принципе и принципе **чисте архитектуре** на идентичан начин као што је показано на вежбама.

Иако је у оквиру спецификације описано како и на који начин треба да се извршава комуникација између ентитета у систему то не значи да класе модели требају у себи имају имплементирану пословну логику!

Класе модели имају само **једну одговорност**, док сервиси у себи имају одговарајућу имплементацију пословне логике или **интерфејсе других сервиса** чије се конкретне имплементације **инјектују кроз конструктор** сервиса.

Тестовима је неопходно покрити минимално **3 базична** случаја, **7 граничних** случајева и **5 произвољних** случајева.

Ентитете није обавезно уносити кроз конзолу, већ је могуће користити помоћне методе које креирају објекте ентитета са насумичним вредностима.

Потребно је имати већ спремне тест податке у апликацији.