SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Student Studentić

RPG IGRA SA ZODB I PYGAME

PROJEKT

BAZE PODATAKA

Varaždin, 2026.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Student Studentić

Matični broj: 12345678

Studij: Informacijski i poslovni sustavi

RPG IGRA SA ZODB I PYGAME

PROJEKT

Mentor:

prof. dr. sc. Dr. sci. Imetor Prezimento

Varaždin, siječanj 2026.

Student Studentić

Izjava o izvornosti

Izjavljujem da je ovaj projekt izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI Radovi

Sažetak

Ovaj projekt prikazuje razvoj RPG igre korištenjem objektno-orijentirane baze podataka ZODB i PyGame frameworka za grafiku. Aplikacija demonstrira perzistenciju kompleksnih Pythonovih objekata, upravljanje stanjem igrača, te integraciju baze podataka s grafičkim sučeljem. Igra uključuje sisteme za: kretanje igrača, sustav zdravlja, inventar stavki, te trajno pohranjevanje podataka igrača u ZODB-u. Sve operacije koriste ACID transakcije kako bi se osigurala konzistencija podataka.

Ključne riječi: ZODB; objektne baze podataka; PyGame; RPG; Python; perzistencija; transakcije; okidači

Sadržaj

1. Uvod

Ovaj projekt je izrađen kao dio kolegija Baze podataka. Demonstrira prednosti objektnih baza podataka u razvoju igara, fokusirajući se na ACID svojstva, transakcije i transparentnu perzistenciju objekata.

Projekt je Role-Playing Game (RPG) igra razvijena u 2D sa osnovnim mehanikama. Igrač kontrolira likove koji se mogu kretati, primati štetu, i skupljati predmete iz svojega inventara. Igra je dizajnirana kako bi demonstrirala kako Zope Object Database (ZODB) efektivno pohrani kompleksne objekte karakteristične za igre.

Igra posjeduje sljedeće mehanike:

1. **Kretanje** – Igrač se kreće tipkama A (lijevo) i D (desno)
2. **Sistem Zdravlja** – Igrač ima 100 HP, pri svakom šteti se smanjuje
3. **Okidač za Porazu** – Kada HP padne na 0, igrač se automatski proglašava poraženim
4. **Inventar** – Igrač može prikupljati stavke
5. **Perzistencija** – Svi podaci se trajna sprema u ZODB

Za ovu aplikaciju, ZODB je odličan izbor jer omogućava da se kompleksni Python objekti (kao što je igrač s inventarom) pohranjuju direktno bez potrebe za mapiranjem na SQL tablice. Primjerice, igrač ima `PersistentList` za inventar koji sadrži `Item` objekte. Objekti se automatski prate u ZODB-u nakon naslijeđivanja `Persistent` klase.

2. Teorijski okvir

2.1. Objektno-Orijentirane Baze Podataka

Objektno-orijentirana baza podataka (Objektno-orijentirana baza podataka (OOBP)) je sustav koji pohrani objekte direktno kao entitete baze podataka, bez potrebe za prevodom u relacijski format. Za razliku od relacijskih baza koje koriste tablice i SQL, OOBP čuva Python objekte kako su kreirani [1].

Osnovna svojstva usporedbe ZODB-a i relacijskih baza podataka prikazana su u tablici 1.

Tablica 1: Usporedba ZODB i Relacijskih BD

Kriterij	**ZODB**	**PostgreSQL**
Osnovni entitet	Python objekt	Tablica/Redak
Upiti	Python kod	SQL
Relacije	Direktne reference	Strani ključevi
Kompleksnost	Jednostavna	Kompleksna za objekte
Tipski sustav	Python tipovi	SQL tipovi

2.2. ACID Svojstva

ACID predstavlja četiri svojstva koja osiguravaju pouzdanost transakcija u bazama podataka [2]:

- **Atomarnost (Atomicity)**: Transakcija se ili kompletan izvršava ili se uopće ne izvršava.
- **Konzistentnost (Consistency)**: Baza je uvijek u konzistentnom stanju.
- **Izolacija (Isolation)**: Konkurentne transakcije ne vide međusobne promjene dok se ne commitaju.
- **Trajnost (Durability)**: Kada je transakcija commitana, sprema se trajno.

2.3. ZODB Specifičnosti

Objekti koji se mogu pohraniti trebaju naslijediti klasu `Persistent`. Transakcije se koriste za grupiranje više operacija. Za spravljanje kolekcija u ZODB-u koristi se `PersistentMapping` i `PersistentList` umjesto običnih Python rječnika i listi [3].

3. Model Baze Podataka

Struktura podataka u igri sastoji se od glavnog korijenskog objekta (root) koji sadrži mape za igrače, tablicu najboljih rezultata i stanje svijeta.

3.1. Klase i Svojstva

3.1.1. Player Klasa

Igrač je glavni lik kojega kontrolira korisnik.

```
class Player(Persistent):
    def __init__(self, name):
        self.name = name
        self._hp = 100
        self.x = 400
        self.y = 300
        self.inventory = PersistentList()
        self.status = "Aktivan"
```

Isječak koda 1: Definicija Player klase

3.1.2. Item Klasa

Stavka je predmet koji igrač može nositi u inventaru.

```
class Item(Persistent):
    def __init__(self, name, power):
        self.name = name
        self.power = power
```

Isječak koda 2: Definicija Item klase

4. Implementacija

4.1. Struktura Projekta

Projekt se sastoji od sljedećih datoteka:

- `src/main.py`: Glavna petlja igre i PyGame logika.
- `src/models.py`: Definicije perzistentnih objekata (`Player`, `Item`).
- `src/database.py`: Upravljanje ZODB vezom i inicijalizacija baze.
- `data/`: Mapa u kojoj se pohranjuju datoteke baze podataka.

4.2. Inicijalizacija ZODB-a

Baza podataka se inicijalizira kreiranjem `FileStorage` i `DB` objekata, te otvaranjem konekcije [3].

```
class GameDB:
    def __init__(self, db_path='data/game.fs'):
        os.makedirs(os.path.dirname(db_path), exist_ok=True)
        self.storage = ZODB.FileStorage.FileStorage(db_path)
        self.db = ZODB.DB(self.storage)
        self.connection = self.db.open()
        self.root = self.connection.root()

        if 'players' not in self.root:
            self.root['players'] = PersistentMapping()
```

Isječak koda 3: Inicijalizacija baze u database.py

4.3. Okidači (Triggers)

Okidač za “HP = 0” je implementiran kao property setter. Kada se HP promijeni, provjerava se je li pao na 0. Ako jest, status igrača se automatski mijenja.

```
@property
def hp(self):
    return self._hp

@hp.setter
def hp(self, value):
    self._hp = max(0, value)
    if self._hp == 0:
        self.status = "Poražen"  # Automatska promjena
    self._p_changed = True  # Javi ZODB-u
```

Isječak koda 4: Okidač - property setter za HP

5. Primjeri Korištenja

5.1. Pokretanje i Igranje

Igra se pokreće naredbom `python src/main.py`. Igrač se kreće tipkama A i D. Pozicija igrača se ažurira na ekranu u realnom vremenu [4].

5.2. Spremanje Stanja

Svaki put kada se igra zatvori, poziva se `transaction.commit()`, čime se svi podaci trajno spremaju u `data/game.fs`. Pri sljedećem pokretanju, podaci se učitavaju točno onako kako su ostavljeni.

6. Zaključak

ZODB se pokazao odličnim izborom za ovu aplikaciju zbog jednostavnosti direktnog spremanja Python objekata bez mapiranja i fleksibilnosti koju pruža. Property setter-i omogućavaju implementaciju logike okidača na prirodan način.

Projekat je uspješno demonstrirao kako ZODB pohranjuje kompleksne Python objekte, korištenje transakcija za atomske operacije, te integraciju baze s grafičkim sučeljem.

Popis literature

[1] C. Beeri i R. Ramakrishnan, „Database research: Achievements and opportunities into the 21st century,” *SIGMOD Record*, sv. 28, br. 1, str. 52–63, 1999.

[2] T. Haerder i A. Reuter, „Principles of transaction-oriented database recovery,” *Computing Surveys*, sv. 15, br. 4, str. 287–317, 1983.

[3] Zope Foundation. „ZODB Documentation. ”adresa: `https://zodb.org`

[4] Pygame Foundation. „Pygame - Getting Started. ”adresa: `https://pygame.org/docs`

Popis tablica