# 2. Scenariji uporabe RAG sustava u visokoškolskom obrazovanju – FER UI asistent

Zbog velike dostupnosti UI asistenata današnji studenti koriste ih svakodnevno. Pri izradi zadataka, domaćih zadaća ili laboratorijskih vježbi studenti često ne promišljaju zadatak samostalno, nego odmah traže objašnjenje i upute od asistenta. Posljedično, sposobnost analize zadatka, razlaganja problema i strukturiranja rješenja postaje sve rjeđa i slabija kod studenata. Navedeni se problemi mogu ublažiti pravilnom uporabom UI asistenata koja potiče aktivno učenje i navodi na samostalno dohvaćanje rješenja problema.

U sljedećim poglavljima prikazani su scenariji uporabe UI asistenata s RAG implementacijom iz perspektive studenta. Nadalje, opisan je model primjene integriranog RAG sustava u FER-ov intranet, te pravila i upute za dobivanje kvalitetnijih odgovora od UI asistenta.

## 2.1. Cilj i pozicioniranje sustava

FER UI asistent je (imaginarni) integrirani, intranet-dostupan UI asistent namijenjen studentima s primarnim ciljem smanjenja napora pri učenju, pronalaženju točnih informacija, izvođenju laboratorijskih vježbi i snalaženju u institucionalnim procedurama. Pri tome osigurava akademski integritet i visoku pouzdanost informacija.

Za razliku od ostalih LLM alata, FER UI asistent dizajniran je kao RAG sustav koji kao izvore informacija koristi samo službene dokumente dostupne na studentovom intranetu. To uključuje materijale iz studentovih aktivnih kolegija, službene dokumente i pravilnike FER-a, obavijesti studentima, akademski kalendar i procedure o međunarodnim razmjenama. U RAG pristupu dokumenti se najprije segmentiraju na manje isječke, pretvaraju u vektorske reprezentacije (embeddings) i pohranjuju u vektorsku bazu. Pri korisničkom upitu sustav dohvaća najrelevantnije isječke te generira odgovor koristeći isključivo dohvaćeni kontekst. Više o tome napisano je u poglavlju ___.

Zbog RAG implementacije sustav značajno smanjuje rizik halucinacija i povećava usklađenost odgovora s lokalnim pravilima i gradivom. Uz svaki odgovor prikazuje se i izvor odgovora (naziv dokumenta, odsječak teksta i link na izvor), kako bi se student mogao dodatno samostalno referencirati na službeni materijal. Sustav primjenjuje kontrolu pristupa, tako da student dohvaća informacije samo iz materijala koje ima pravo vidjeti (npr. vlastiti aktivni kolegiji), dok se administrativni izvori dohvaćaju iz zasebnog skupa službenih dokumenata.

FER UI asistent ima tri glavna načina rada: kolegijski, laboratorijski i administrativni asistent.

## 2.2. Kolegijski asistent

Kolegijski asistent je *chat* sučelje koje omogućuje studentu da postavlja pitanja o svojim kolegijima, a asistent odgovara koristeći isključivo sadržaj repozitorija studentovih aktivnih kolegija. Pregledava obavijesti, materijale, kalendar i nastavne aktivnosti. U slučaju višestrukih

verzija materijala, sustav koristi *metapodatke* (kolegij, tema, tjedan, akademska godina, tip dokumenta) i prema njima filtrira dokumente za pretraživanje.

Njegova primarna namjena je pomoć u učenju i razumijevanju sadržaja kolegija. Konfiguriran je tako da ako ne može pronaći odgovor u materijalima, odgovori „Nije mi dostupna ta informacija u izvorima.“ i proslijedi službeni kontakt asistenta ili profesora na kolegiju.

### 2.2.1. Scenarij uporabe 1: Odgovaranje na pitanja za razumijevanje

***Primjer upita**:*

- „Objasni razliku između X i Y kako je definirano na predavanjima.“
- „Pregledaj moje rješenje i ocijeni ga prema kriterijima zadatka.“
- „Što se traži u zadatku X i gdje u materijalima mogu to pronaći?“

Asistent generira odgovor i navodi referencu na službeni materijal i isječak u kojem je pronašao informaciju.

### 2.2.2. Scenarij uporabe 2: Generiranje pitanja za učenje

Asistent također može generirati pitanja za učenje iz željenog nastavnog materijala uz objašnjenja i reference na materijale.

***Primjer upita*:** „Generiraj mi 10 pitanja iz gradiva logistička regresija iz kolegija strojno učenje 1.“

***Tok izvođenja*:**

→ studentov šalje upit
→ sustav generira pitanja
→ student odgovara na pitanja
→ sustav ocjenjuje odgovore i daje referencu na materijale, predlaže studentu kojem se još dijelu gradiva posvetiti.

### 2.2.3. Scenarij uporabe 3: Navigacija obavezama i rokovima

Konačno, kolegijski asistent ima pristup studentovom kalendaru i rasporedima kolegija. Prema tome, može pomoći studentu u sastavljanju rasporeda učenja, sastaviti pregled njegovih obaveza za tekući tjedan i slično.

***Primjer upita:***

- „Koje sve laboratorijske vježbe moram predati ovaj tjedan?“
- „Koliki je prag bodova na završnom ispitu iz kolegija X?“

Asistent generira odgovor i navodi referencu na službeni materijal i isječak u kojem je pronašao informaciju.

## 2.3. Laboratorijski asistent

Laboratorijski način rada može se koristiti za dijagnosticiranje problema pri rješavanju laboratorijskih vježbi. Cilj ovog načina rada je smanjiti vrijeme izgubljeno na repetitivne probleme kao što su konfiguracija okruženja, spajanje opreme i format predaje.

### 2.3.1. Scenarij uporabe 3: Dijagnostika problema pri izvođenju laboratorijske vježbe

***Pretpostavka***: svaka laboratorijska vježba sadrži dokumentaciju korištenih alata i upute za izvođenje vježbe.

***Primjer upita***: „Izvodim laboratorijsku vježbu 2 iz kolegija Osnove elektrotehnike i nemam signala na osciloskopu.“

Asistent predlaže provjere u prioritetnom redoslijedu (prvo najčešći uzroci problema) i upozorava na sigurnost ukoliko je potrebno.

### 2.3.2. Scenarij uporabe 4: Dijagnostika problema u programskom kodu

Osim dijagnostike tijekom izvođenja laboratorijskih vježbi “uživo”, laboratorijski način rada može se koristiti i kao pomoć pri pisanju programskog koda. Naglasak je pritom na tome da asistent ne isporučuje kompletno programsko rješenje, već vodi studenta kroz proces rješavanja, pomaže u otkrivanju uzroka greške i predlaže testove. Zabranjeno generiranje potpunog rješenja definira se osnovnom konfiguracijom sustava.

***Primjer upita***: „Rješavam laboratorijsku vježbu X iz kolegija Y. Ovdje je moj kod {…}. Kod baca grešku: {…}“

***Tok izvođenja***:

- student šalje upit
- sustav navodi 2-3 hipoteze uzroka i korake izolacije problema
- student pokušava primijeniti korake
- ako problem i dalje nije riješen, sustav daje *hint-ove* za rješavanje problema

Rezultat korištenja ovakvog asistenta pri pisanju bio bi veće razumijevanje koda, bolje testiranje rubnih slučajeva i visoka evaluacija rješenja.

## 2.4. Administrativni asistent

Administrativni asistent namijenjen je odgovaranju na pitanja vezana uz administrativne informacije FER-a: pravilnike, procedure, mobilnosti, upise, rokove, obrasce i slično. On informacije dohvaća isključivo iz službenih administracijskih dokumenata. Sustav je konfiguriran tako da, ako ne može pronaći odgovor u dostupnim izvorima, odgovori: “Nije mi dostupna ta informacija u izvorima.” te priloži službeni kontakt studentske službe FER-a.

### 2.4.1. Primjer uporabe administrativnog asistenta

Tipovi upita administrativnom asistentu:

- Upiti vezani uz pravila studiranja i postupe
- Upute za studentske službe
- Upute za mobilnosti i međunarodnu razmjenu
- Upute za navigaciju po službenim izvorima

Iako je samo predstavljena ideja, ovakav UI RAG asistent uvelike bi olakšao svakodnevicu svim studentima FER-a i povećao akademski integritet studentskih radova s *online* predajom.

U sljedećem poglavlju dane su upute za pisanje učinkovitih i jasnih upita bilo kojem UI asistentu.

### 2.5 Pisanje upita

Usprkos učestalom korištenju UI asistenata, mnogi studenti idalje ne znaju kako treba izgledati dobar upit. Frustrirani šalju upite koji ne daju asistentu dovoljno uputa i zaključe da asistent ipak nije toliko dobar koliko su mislili.

Neke od najčešćih grešaka pri postavljanju upita su:

- preopćeniti upiti
- upiti bez izlaznog formata
- upiti bez ograničenja izvora (model nadopunjuje informacije iz svog općeg znanja)
- *copy/paste* teksta bez cilja.

Kako bi se povećala efektivnost i preciznost upita, u nastavku su navedene upute za sastavljanje učinkovitog upita.

#### 2.5.1. Formula za dobar upit

Generička forma kojom se sastavlja većina upita sastoji se od 5 komponenti:

1. ***Cilj***: koja se akcija traži od sustava (objasni / generiraj / usporedi / provjeri).
2. ***Kontekst***: područje znanja, razina razumijevanja korisnika
3. ***Ograničenja***: koristi samo priložene materijale / navedi pretpostavke / bez izmišljanja
4. ***Format izlaza***: lista, tablica, koraci, pseudokod, LaTeX…
5. ***Provjera***: generiranje testova, rubnih slučajeva, ili “navedi gdje si nesiguran”.

Bez ovakve forme, UI asistenti će vratiti općeniti odgovor o temi upita koji ne mora sadržavati informacije koje je korisnik htio dobiti.

#### 2.5.2. Sprječavanje halucinacija

Iako se nikada ne može u potpunosti spriječiti haluciniranje, preciziranjem u upitu može se značajno smanjiti vjerojatnost halucinacije odgovora. Neki od primjera rečenica za ubaciti u upit u svrhu sprječavanja halucinacija su:

- „Ako informacija nije u kontekstu, reci 'nije u izvorima'.“
- „Za svaku tvrdnju navedi citat i link na izvor.“
- „Navedi pretpostavke i stupanj sigurnosti u odgovor.“

### 2.5.3. Iterativni upiti

Još jedna metoda za osiguravanje kvalitetnog i ispravnog odgovora je postavljanje dodatnih upita kojima asistent propituje svoje odgovore. Jedan od uzoraka postavljanja upita može biti:

*Odgovor → Kritika → Ispravljena verzija*

***Primjer:*** Korisnik šalje sljedeće upite:

1. „Napiši rješenje zadatka: {…}“
2. „Recenziraj rješenje kao strogi asistent.“
3. „Ispravi rješenje prema recenziji.“

Kako bi studenti lakše primijenili prethodno navedenu formulu i smanjili broj nepreciznih ili preopćenitih upita, u nastavku su prikazani predlošci koji se mogu izravno prilagoditi konkretnom kolegiju, temi i zadatku. Preporuka je predloške također koristiti iterativno.

### 2.5.4. Predlošci studentima za pisanje upita

Predlošci su osmišljeni tako da eksplicitno navode cilj, kontekst i format izlaza te, po potrebi, uključuju ograničenja izvora i zahtjev za provjerom točnosti. Time se povećava vjerojatnost da asistent vrati odgovor koji je istovremeno koristan i pouzdan.

1. **Objašnjenje pojma i provjera razumijevanja**
   “Objasni mi **[*pojam*]** na razini studenta FER-a (X. godina).
   Daj: (1) intuitivno objašnjenje, (2) formalnu definiciju, (3) primjer iz prakse, (4) 5 brzih pitanja za samoprovjeru s odgovorima.”
2. **Sokratski način**
   “Glumi demonstratora. Nemoj mi odmah dati rješenje. Postavljaj mi pitanja i daj *hintove* dok ne dođem do rješenja zadatka: **[*tekst zadatka*]**.”
3. **Generiranje testnih slučajeva u programskom kodu**
   “Za funkciju/spec: **[*opis*]** generiraj testove: normalni slučajevi, rubni slučajevi, slučajevi greške. Za svaki napiši očekivani ishod.”