

Statut Univerziteta Džemal Bijedić

Član 175.

(Povrede obaveza)

(2) Kao teže povrede obaveza smatraju se naročito:

...

(c) korištenje nedozvoljenih sredstava na ispitu ili prilikom izrade projekata, seminarskih radova, praktičnih radova i sl.;

...

Član 176.

(Disciplinske mjere)

(2) Mjera isključenja može se izreći samo za težu povredu obaveze, i to u trajanju od 1 (jedne) do 3 (tri) akademske godine.

Član 177.

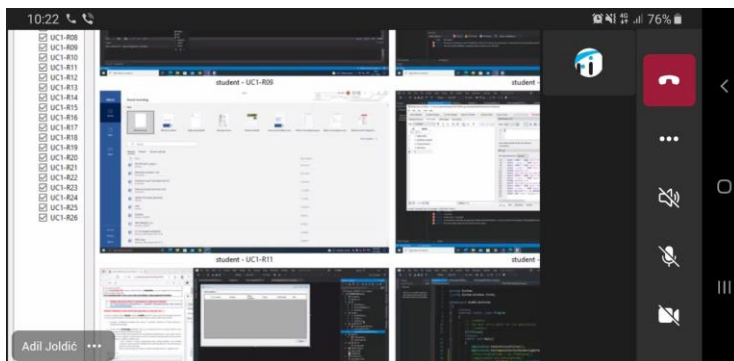
(Postupak utvrđivanja povrede obaveza)

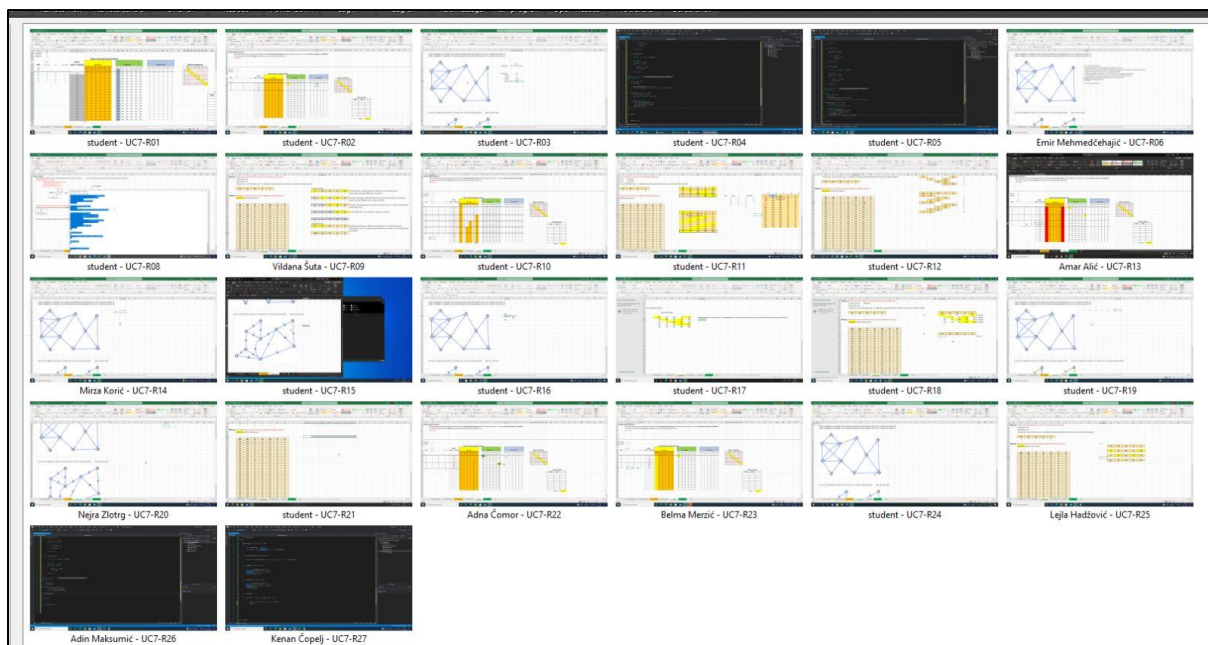
...

(2) Postupak iz prethodnog stava se može pokrenuti u roku od 6 (šest) mjeseci od dana saznanja za povredu obaveze i počinioca, a najkasnije u roku od 12 (dvanaest) mjeseci od dana kada je povreda učinjena.

...

- Ispitne zadatke nije dozvoljeno slikati niti slati na email tokom ispita.
- Sadržaj ekrana (monitora) je pod video nadzorom preko namjenskog softvera.
- Sadržaj ekrana se snima na server.
- Sadržaj ekrana uživo prati dežurno nastavno osoblje u učionici i izvan nje (preko računara ili mobitela).
- Mrežni promet je pod nadzorom preko firewall servera.
- Prati se broj aktivnih windows procesa.
- Prilikom uploada na FTP server logira se IP adresa računara i vrijeme.





RS1 - Praktični ispit - Važne napomene za NPM komande, VS code i Webstorm

NPM komande u učionici (PowerShell ne radi)

U učionici npm komande ne rade iz PowerShell terminala (Windows PowerShell, PowerShell u VS Code, WebStorm PowerShell), jer je blokirano pokretanje skripti (npm.ps1).

Rješenje: u istom terminalu prvo prebacite na CMD, pa tek onda pokrećite npm:

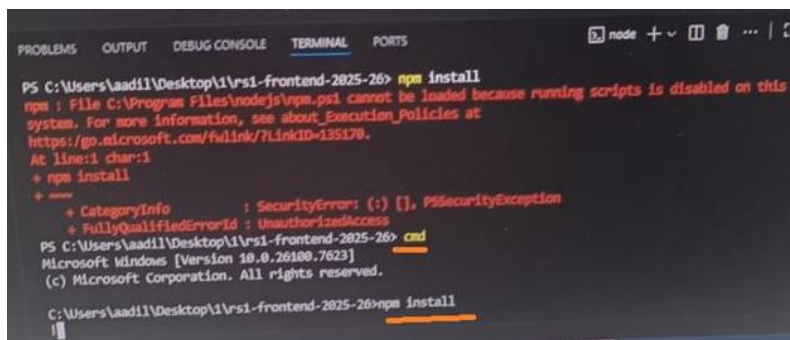
1. U PowerShell-u upišite:

cmd

2. Pritisnite Enter

3. Zatim pokrećite npm komande, npr.:

npm install
npm start



ng komande

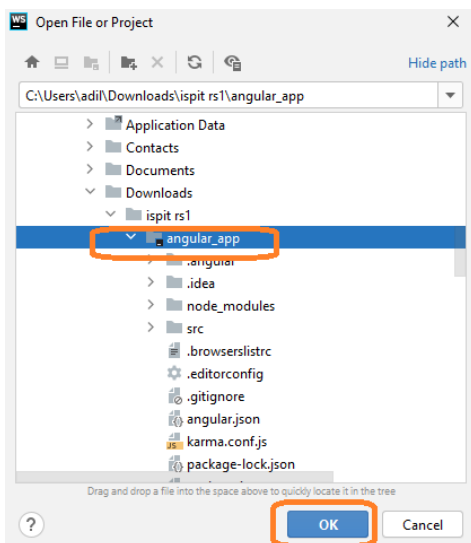
- Za generisanje novih Angular komponenti:
 - Probajte standardno: **ng g c NazivKomponente**
 - Ako to ne radi (**ng** nije dostupan u PATH-u), koristite: **npx ng g c NazivKomponente**
- Napomena: **npx** pokreće Angular CLI iz lokalnih **node_modules** (ili ga privremeno preuzme).

Frontend razvoj: VS Code ili WebStorm

- Za Frontend možete koristiti VS Code ili WebStorm.
- VS Code je već instaliran i spreman za korištenje na učioničkim računarima.
- WebStorm: na većini učioničkih računara licenca nije aktivna. Ko želi koristiti WebStorm, treba:
 1. Kući kreirati JetBrains account (bilo koja email adresa, ne mora biti edu.fit.ba)
 2. U WebStorm-u odabrati opciju **Non-commercial** i aktivirati preko svog naloga (nema potrebe ići na studentsku licencu)
 3. Na učioničkom računaru se prijaviti tim nalogom.

Upute

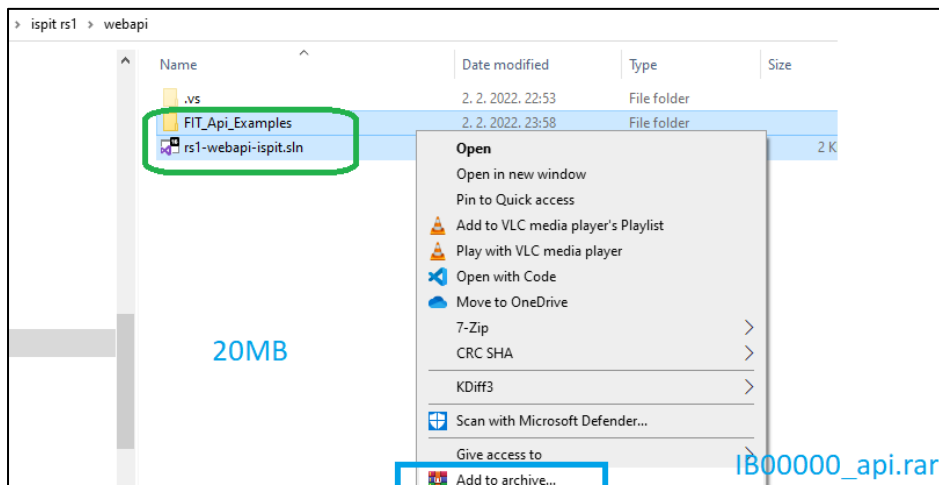
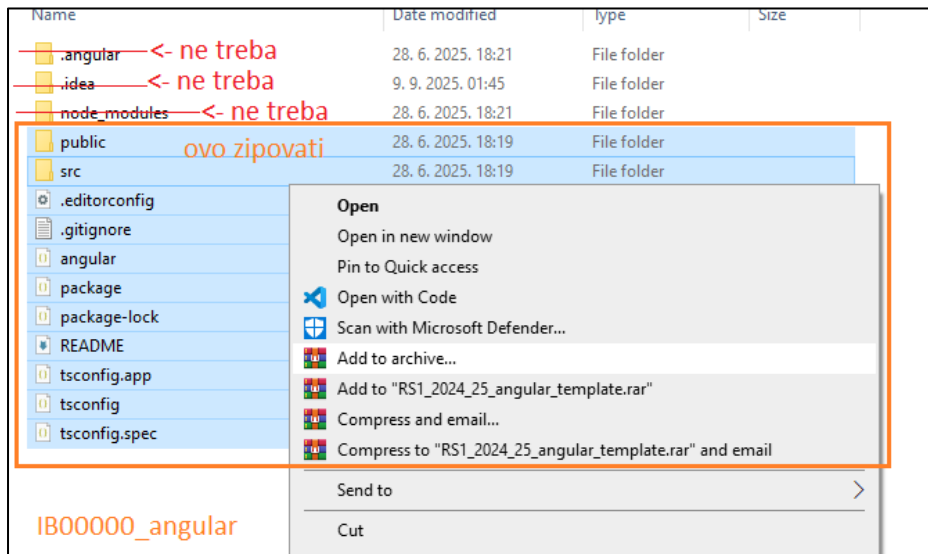
- a) Otvoriti Visual Studio **2022**
- b) Otvoriti SqlServer Managment Studio, konektovati se na sql server i napraviti novu bazu podataka u čijem nazivu je broj indeksa.
 - sql server adresa: 10.10.10.18
 - sql server integrated security: false
 - username: sa
 - password: test
- c) Preuzeti projekat sa **rs1-2026-01_26-stari-silabus-projekat.zip** sa FTP-a:
 - a. Podaci za FTP
 - i. Adresa: **ftp.fit.ba**
 - ii. Username: **student_aj**
 - iii. Password: **student_aj**
 - b. RAR kopirati na desktop i otpakovati
- d) **Otvoriti u WebAPI u VS-u**
 - a. Prepraviti connection string (**upisati broj indeksa za dbname**)
 - b. Izvršiti migracije (update-database) te pokrenuti projekat
 - c. Generisati testne podatke preko akcije **TestniPodaci/Generisi**



Upload

Potrebno je uploadovati **dva rar ili zip fajla** (web api i angular projekat).

- IB00000_stari-angular.rar
- IB00000_stari-api.rar



Trajanje ispita

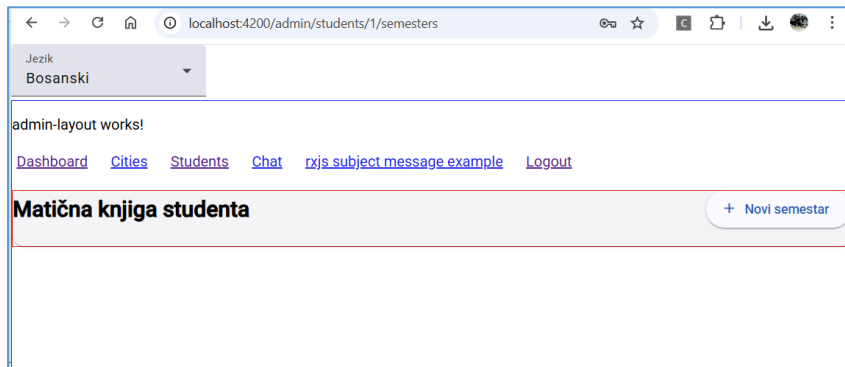
2h

Napomena 1: Bodovat će se samo u potpunosti funkcionalna rješenja, a ne polovična.

Napomena 2: Naziv solution fajla preimenovati o broj indeksa!

Napomena 3: Svi zadaci se rade na formi "Matična knjiga – semestri"

Zadatak A: Lista semestara + status filter (Active / Deleted / All) + Delete/Restore



U početnom radnom projektu postoje rute i navigacija prema ekranu Semestri/Matična knjiga, ali funkcionalnost liste i forme nije implementirana. Potrebno je implementirati matičnu knjigu koja sadrži listu upisa (semestra/godina) za odabranog studenta.

A.1 Backend zahtjevi (endpoint za listu)

Implementirati endpoint za prikaz semestara upisanih za konkretnog studenta:

- GET /students/{studentId}/semesters/filter

Endpoint mora podržati:

- paginaciju (pageNumber, pageSize – kao i ostali filter endpointi u projektu)
- pretragu q (tekstualni upit)
- novi parametar statusa: status=active|deleted|all

Status filter logika

1. active → vraća samo zapise koji nisu obrisani
2. deleted → vraća samo obrisane (soft-deleted) zapise
3. all → vraća sve (i aktivne i obrisane)

A.2 Response polja (obavezno)

Svaki red u listi mora imati barem:

- id
- academicYearDescription (tekst iz AcademicYear)
- studyYear (godina studija)
- enrollmentDate (datum upisa)
- isRenewal (obnova)
- tuitionFee (cijena)
- isDeleted (boolean)

A.3 Frontend zahtjevi (UI)

U StudentSemestersComponent implementirati:

- tabelu sa kolonama minimalno:
 - Akademska godina
 - Godina studija
 - Datum upisa
 - Obnova
 - Cijena
 - Actions
- kontrolu statusa (radio / button-toggle / select):
 - **Active / Deleted / All** (default Active)

A.4 Actions kolona (Delete vs Restore)

- Ako je red **aktivan** (isDeleted=false) → prikazati **Delete** dugme (uz confirm dialog)
- Ako je red **obrisan** (isDeleted=true) → prikazati **Restore** dugme (uz confirm dialog)
- U režimu **All** obrisane redove vizuelno označiti (npr. css klasa obrisano, badge "DELETED", ili sl.)

A.5 Soft delete / restore endpointi

Implementirati:

- DELETE /student-semesters/{id} → soft delete zapisa

- POST /student-semesters/{id}/restore → restore zapisa

Očekivano ponašanje (check)

- U Active/All režimu mogu obrisati upis → nestaje iz Active, pojavljuje se u Deleted.
- U Deleted/All režimu mogu restore → nestaje iz Deleted, pojavljuje se u Active.
- Nakon Delete/Restore lista se osvježi i **ostaje u trenutno odabranom status režimu.**

Zadatak B: Novi upis (Reactive form) + validacije + automatski izračun

U StudentSemestersNewComponent implementirati dodavanje novog upisa zimskog semestra / godine.

B.1 Backend zahtjevi (entity + migracija)

Potrebno je dodati novu tabelu (nova entity klasa + migracija). Predložena polja:

Obavezna polja:

- StudentId (FK)
- AcademicYearId (FK na AcademicYear)
- StudyYear (int)
- EnrollmentDate (DateTime)
- TuitionFee (float/decimal)
- IsRenewal (bool)
- EvidentiraoKorisnikId (FK na korisnika koji evidentira)
- IsDeleted (bool) – zbog soft delete/restore na semestrima

(Pošto projekat koristi tenant model: entity treba pratiti isti pristup kao Student – tj. tenant-specific tabela.)

B.2 Endpoint za dodavanje

Implementirati:

- POST /students/{studentId}/semesters

Request treba sadržavati:

- academicYearId
- studyYear
- enrollmentDate

Backend treba upisati:

- evidentiraoKorisnikId iz logovanog korisnika (ne šalje se s forme)
- tuitionFee i isRenewal izračunate po pravilima (vidi ispod)

B.3 Pravila za obnova i cijenu (obavezno)

Na osnovu istorije upisa tog studenta:

- isRenewal = true ako student upisuje **istu godinu studija** kao prethodno (raniji aktivni upisi)
- isRenewal = false ako je nova godina studija

Cijena (tuitionFee):

- 400 ako student upisuje istu godinu studija **prvi put**
- 500 ako student upisuje istu godinu studija **drugi put**
- inače 1800

Bitno: pravila se moraju primijeniti dosljedno. Frontend može prikazivati izračun, ali backend mora biti tačan.

B.4 Frontend forma (ReactiveForms)

Implementirati Angular Reactive Form sa poljima:

- Datum upisa (date picker)
- Godina studija (number)
- Akademska godina (combo) – popuniti iz API-a
- Obnova (readonly/disabled)
- Cijena (readonly/disabled)

Akademska godina lookup

AcademicYearLookupEndpoint za listu za combo je već implementiran u BE i FE:

- GET /academic-years/lookup

B.5 Validacije

Obavezno:

- Backend validacija: spriječiti unos iste akademske godine više puta za istog studenta (za aktivne upise).
- Ako postoji obrisani upis za istu akademsku godinu, korisnik treba koristiti Restore, a ne kreirati novi upis
- Frontend validacija: ne dozvoliti submit ako nisu unesena obavezna polja.

Očekivano ponašanje (check)

- Nakon Save, vraća na /admin/students/:id/semesters i lista prikazuje novi upis.
- Ne može dodati isti academicYearId dvaput za istog studenta (aktivno stanje).

Zadatak C: RxJS: Pretraga + debounce + “pametan” q (300ms)

Na listi semestara dodati input za pretragu (q).

C.1 Zahtjev

Trenutno (kad završite zadatak B) najlakše je zvati API na svaki keyup.

Međutim, u ovom zadatku API se smije pozvati tek kad korisnik prestane kucati **300ms**.

C.2 Obavezno (RxJS dio)

1. U StudentSemestersComponent napraviti Subject<string> (npr. search\$)
2. U ngOnInit() napraviti subscription:
 - search\$.pipe(...) mora sadržavati:
 - debounceTime(300)
 - distinctUntilChanged()
 - pretvaranje u mala slova (npr. map(x => x.toLowerCase()))
 - slanje q samo ako je dužina stringa > 3, inače šalji q = '' (ili ne šalji q)
3. applyFilter() (ili handler inputa) više ne smije direktno zvati fetch..., nego samo:
 - this.search\$.next(value)

C.3 Debug logovi

- Prije slanja requesta: ispisati q u konzolu
- Nakon response-a: ispisati broj recorda (npr. totalCount)

C.4 Paginacija

- Kad se promijeni search, reset paginaciju na prvu stranicu.
- Paginator i dalje radi normalno (page change učitava sljedeću stranicu sa zadnjim status i zadnjim q).

Očekivano ponašanje (check)

- Dok korisnik kuca “m o s t a r”, API se ne smije zvati 6 puta, nego 1 put nakon pauze.
- Status filter + search rade zajedno.