1. Svaki resurs (studenti, profesori, predmeti ...) neka budu zapisani datoteke (studenti.txt, profesori.txt)

Stvorio sam 4 datoteke Studenti.txt,Profesori.txt,Predmeti.txt,Ocjene.txt i pazio sam da se ni jedno prezime ne ponavlja pa po tome mozemo povezivati i traziti na osnovu prezimena,u koliko redni brojevi na pocetku budu stvarali probleme mozemo ih lako ukloniti,datoteku Ocjene.txt sam ostavio praznu pa mozemo u nju spremati ocjene i dodijeliti ima neki redni broj na osnovu toga ce nam u toj datoteci trebati redaka onoliko koliko iznosi umnozak broja predmeta sa brojem studenata pretpostavivsi da je svaki od navedenih studenata upisao svaki od navedenih predmeta

1. Format zapisa datoteke smislite sami.
2. Logika upravljanja resursima neka bude odvojena u modulima (studenti.h, studenti.c; profesori.h, profesori.c ....)

Ako napravimo header file ili nesto.h i ukljucimo to u .c sa #include „nesto.h“ onda ne moramo stavljati prototipe funckija,

1. Svaki modul nudi sljedeće osnovne funkcije (primjer sa studentima):
   * int save(Student \*student) - sprema pojedinačni resurs sa generiranim ID-jem u bazu podataka (ID mora biti jedinstven i mora biti izgeneriran sa strane baze podataka)
   * int saveAll(StudentList \*studentList) - isto kao save, samo sprema listu resursa
   * Student \*findById(ID id) - dohvati zapis po ID-u
   * StudentList \*findAll() - dohvati sve studente
   * StudentList \*findAllBy<Something>() - dohvati sve po nekom atributu (ime, prezime itd.)
   * StudentList \*findBetween<Something>And<Something>() - dohvati sve po nekom atributu između.
   * bool existById(ID id) - vidi jel postoji zapis po ID-u (stdbool.h)
   * int deleteById(ID id) - izbriši zapis po ID-u
   * int deleteAll() - izbriši sve zapise
2. Svaka funkcija radi sljedeće:
   * Učitaj podatke u AVL binarno stablo pretraživanja (ključ: ID, vrijednost: podatak/struktura).
   * Ukoliko pretražujete po nekom atributu, stvoriti AVL stablo po principu (ključ: atribut, vrijednost: podatak/struktura)
   * Zatvoriti datoteku
   * (Ono što funkcija treba zapravo obaviti)
   * Pobrisati alociranu memoriju
   * Vratiti rezultat
3. Vezati resurse međusobno s pokazivačima (Smislite sami kako ih povezati i čitati iz datoteka)
4. Napraviti program u kojem korisnik unosi primjerom:

Akcija -> find studenti 13

Odgovor -> id: 13, first\_name: Ante, last\_name: Milić

Akcija -> find studenti

Odgovor -> (isto kao prvi samo u nizu)

8.Za početak pročitajte malo o REST arhitekturi.

Video o REST(Representational State Transfer) strukturi:

<https://www.youtube.com/watch?v=SLwpqD8n3d0>

9.Prezentirat ćete rješenje na način da ćete imati live demo prezentaciju gdje ćete pokazati što Vaš sve program može.