

Harjoitustyö raportti

Sain koulusta tehtäväksi rakentaa hotellivarausjärjestelmän, joka koostuu sekä C++ backend-ohjelmasta että Next.js frontend-sovelluksesta. Alusta asti halusin tehdä kunnolla töitä ja tavoitteelin parasta arvosanaa, joten lähdin suunnittelemaan järjestelmän niin, että se täyttäisi kaikki vaatimukset. Se antoi minulle selkeän suunnitelman siitä, mitä ominaisuuksia minun piti toteuttaa ja miten järjestelmän pitäisi toimia.

1. Projektin työosuus

Aloitukseen pääsin tekemällä pääfunktioon while-loopin, jossa käyttäjä voi valita halutun toiminnon valikosta. Valikko näyttää neljä eri vaihtoehtoa ja ottaa käyttäjältä numeerisen syötteen.

Varausjärjestelmää rakentaessani tarvitsin ensin luoda hotellin pohjan. Tein funktion, joka generoi satunnaisten huoneiden määrän välillä 40-70 ja varmistaa, että se on parillinen. Rakennin huoneille struktuurin, jossa oli kaikki tarvittavat tiedot (numero, tyyppi, varauksen tila, asiakkaan nimi, varausnumero, alennus). Loin Hotelli-nimisen struktuurin, joka sisältää taulukon huoneille sekä huoneiden määrän. Tallennin kaikki nämä tiedot JSON-tiedostoon, jotta varaukset eivät katoa ohjelman sulkeutuessa. Ohjelmalle kirjoitin myös logiikan, joka tarkistaa käynnistyessä, löytyykö tiedosto jo olemassa, ja sen jälkeen joko lataa vanhat tiedot tai alustaa kokonaan uuden hotellin.

Seuraavaksi lähdin toteuttamaan varsinaisia ominaisuuksia: varauksen tekemisen, varausten hakemisen ja hotellin tilanäkymän. Lisäsin myös kunnollisen syötteen tarkistuksen, joka varmistaa, että käyttäjä syöttää oikeanlaisia arvoja eikä ohjelma kaadu virheellisiin syötteisiin. Lopussa paransin vielä käyttöliittymää ja siivosin koodia. Aika oli välillä tiukilla, joten joitain ominaisuuksia en ehtinyt lisätä, esimerkiksi case-insensitive hakua varauksia etsiessä.

Frontend-puolella käytin Next.js 16:aa, TypeScriptiä ja Tailwind CSS:ää rakentaakseni modern- ja responsiivisen käyttöliittymän. Käytin myös Framer Motionia animaatioille ja Shadcn/UI-komponentteja UI-elementeille. Ennen toteutusta suunnittelin käyttöliittymän rakennetta Figmaassa, jotta saisin selkeän vision siitä, miten sovelluksen pitäisi näyttää ja toimia. Tämä auttoi minua tekemään johdonmukaisen ja käyttäjäystävällisen designin.

2. Projektin työosuus

Ohjelma alkaa pääfunktioista jossa:

1. Alustetaan satunnaislukugeneraattori (srand)
2. Kutsutaan alustaHotelli() joka arpoo huoneiden määrän (40-70, parilliset)
3. Aloitetaan while-silmukka, joka näyttää päävalikon ja käsittelee käyttäjän valinnat

Päävalikossa käyttäjä voi valita:

- 1 = Uusi varaus
- 2 = Etsi varaus
- 3 = Hotellin tila
- 4 = Lopeta

Varauksen tekeminen (valinta 1):

- Käyttäjä syöttää nimensä
- Valitsee huonetyypin (1 = yksiö, 2 = kaksio)
- Syöttää yöiden määrän
- Ohjelma arpoo varausnumeron (10000-99999, tarkistetaan uniqueness)
- Ohjelma arpoo alennuksen (0%, 10%, tai 20%)
- Näytetään varauksen vahvistus
- Päivitetään JSON-tiedosto

Varausten haku (valinta 2):

- Voi etsiä nimellä tai varausnumerolla
- Näyttää kaikki varatun huoneen tiedot
- Voi tehdä uusia hakuja

Hotellin tila (valinta 3):

- Näyttää statistiikat: vapaita huoneita, varattuja, hinnat

Lopetus (valinta 4):

- Sulkee ohjelman

JSON formaatti on seuraava:

```
{  
  "huoneidenMaara": 56,  
  "vapaanaHuoneita": 54,  
  "varatuita": 2,  
  "yksioHinta": 100,  
  "kaksioHinta": 150,  
  "huoneet": [  
    {  
      "numero": 1,  
      "tyyppi": 1,  
      "varattu": true,  
      "asiakas": "Nimi",  
      "varausNumero": 28450,  
      "alennus": 0  
    }  
  ]  
}
```

3. Projektin työosuus

Opin tästä projektista paljon käytännön asioita C++:sta. Merkittävin oppitunti oli se, että rakennellun koodin kirjoittaminen vaatii etukäteissuunnittelua. Aluksi minulla oli hieman vaikeuksia ymmärtää, miten struct-rakenteet toimivat ohjelman suuremmassa mittakaavassa, mutta projektin edetessä kävi selväksi, kuinka ne järjestävät koodia paljon paremmin kuin satunnaiset muuttujat.

Debugging-aidot kehittyivät myös merkittävästi. Kun ohjelma ei toiminut odotetusti, piti opetella lukemaan error-viestejä ja jäljittämään ongelmat systemaattisesti. Useissa tapauksissa ongelma oli pieni logiikan virhe, joka tuli ilmi vain testaamalla erilaisia käyttäjäskenaarioita.

JSON-tiedostojen käsittely oli myös uutta ja haastavaa, mutta lopuksi sain sen toimimaan kunnolla. Nyt ohjelma pystyy tallentamaan ja lataamaan tietoja ilman ongelmia. Käyttöliittymän kehittäminen sekä C++:ssa että Next.js:ssä paransi myös ymmärrystäni siitä, miten tärkeää on tehdä ohjelmasta käyttäjäystävällinen ja johdonmukainen.

Vaikka aikaa oli rajoitetusti, lopputulos on mielestäni hyvä. Ohjelma toimii vakaasti ja täyttää kaikki vaatimukset. Koodi on selkeää ja hyvin strukturoitua. Frontend on nykyaikainen ja responsiivinen. Jatkossa voisin parantaa hakutoimintoa tekemällä siitä case-insensitiivisen ja lisätä vielä muita ominaisuuksia, kuten varauksen muokkaamisen sekä backend- että frontend-puolella.