Slide 1 și 2

Lucrarea mea se intitulează «Aplicație pentru planificare inteligentă utilizând tehnici de AI» și a fost realizată sub coordonarea doamnei S.I. dr. ing. Mihaela Țiplea.

Slide 3 — Scopul și obiectivele lucrării

Scopul acestui proiect a fost automatizarea procesului de generare a orarului universitar, pentru a reduce timpul necesar și a elimina cât mai mult erorile umane din planificare.

Am ales să fac acest lucru prin integrarea unei soluții AI, pe care am comparat-o cu un algoritm clasic, tocmai pentru a vedea clar care aduce mai multe beneficii.

Iar rezultatul final a venit sub forma unei aplicații web moderne și adaptabile, potrivită nevoilor unei universități.

Slide 4 — Descrierea aplicației și a problemei

Am pornit de la problema realizării manuale a orarelor, un proces consumator de timp și predispus la erori.

Soluția a fost să creez o aplicație web modernă, care să gestioneze complet profesorii, sălile, grupele și regulile academice.

În plus, am vrut să pot compara două moduri de generare a orarului: unul clasic, cu reguli scrise în Python, și unul bazat pe AI, astfel încât să pot vedea concret avantajele și eficiența celor două abordări.

Slide 5 — Tehnologii utilizate

Pentru dezvoltare am folosit un stack tehnologic modern:

- Frontend-ul este realizat în React, folosind JavaScript și JSX, care îmi permite să structurez aplicația în componente reutilizabile și să am un cod mult mai organizat. Pentru partea vizuală, am integrat Bootstrap, astfel încât interfața să fie modernă și responsive, adaptată la orice dispozitiv.
- Comunicarea dintre frontend și backend se face prin Fetch API, prin care frontend-ul trimite cereri HTTP către serverul Flask, fie pentru a prelua date, fie pentru a le salva.
- Backend-ul rulează în Flask pe Python, unde am implementat logica aplicației sub forma unui API REST.
- Datele sunt stocate într-o bază de date MySQL, cu tabele pentru profesori, săli, grupe, reguli și orare, legate prin chei externe.
- Pentru generare, am două metode: una clasică în Python, și una bazată pe AI cu GPT-4, unde folosesc prompturi detaliate care descriu toate constrângerile academice.

- Rezultatul final poate fi exportat rapid în PDF şi Excel, folosind ReportLab, PyPDF2 şi xlsxwriter.

Slide 6 — Arhitectura şi funcţionarea sistemului

Aplicaţia a fost dezvoltată pe o arhitectură pe trei niveluri: interfaţa grafică realizată în React, comunică printr-un API REST implementat în Flask, iar datele sunt stocate într-o bază de date MySQL, structurată cu tabele pentru profesori, săli, grupe, reguli şi orare.

Fluxul principal începe cu introducerea datelor, continuă cu generarea orarului, care se poate face fie prin AI, utilizând GPT-4, fie printr-un algoritm clasic scris în Python. Ulterior, orarul trece printr-un proces de validare automată, unde sunt verificate structura JSON şi respectarea tuturor constrângerilor academice.

La final, rezultatul poate fi exportat direct în PDF sau Excel, fiind astfel gata pentru distribuire.

Metodologia este una hibridă, care integrează atât inteligenţa artificială, cât şi soluţia clasică, pentru a putea evalua în mod obiectiv avantajele şi limitările fiecărei abordări.

Slide 7 — Principiul funcţionării aplicaţiei

Aplicaţia se deschide cu o pagină de bun venit, unde utilizatorul poate să se logheze dacă are deja cont sau să îşi creeze un cont nou.

După autentificare este redirecţionat în dashboard, care este practic centrul aplicaţiei.

Slide 8 — Principiul funcţionării aplicaţiei

În prima fază, introducem datele esenţiale. Aici se definesc grupele şi subgrupele pe fiecare an, se înregistrează sălile disponibile, iar la profesori sunt incluse disciplinele predate şi disponibilitatea lor săptămânală. Toate aceste date sunt esenţiale pentru generarea ulterioară a orarului.

Slide 9— Principiul funcţionării aplicaţiei

În continuare, avem setarea regulilor academice. Utilizatorul poate introduce direct din interfaţă constrângerile — de exemplu cursurile comune pe an, seminariile pe grupă, laboratoarele pe subgrupă, sau pauza obligatorie miercuri de la 14 la 16. Sau, în cazul generării cu AI, poate personaliza promptul JSON în funcţie de nevoile instituţiei.

Slide 10 — Principiul funcţionării aplicaţiei

După ce toate datele şi regulile sunt setate, putem genera efectiv orarul. Utilizatorul alege dacă vrea să folosească AI-ul GPT-4, care adaptează rapid orice reguli noi doar prin schimbarea promptului, sau algoritmul Python clasic, care are reguli hardcodate.

Slide 11— Principiul funcţionării aplicaţiei

La final, orarul este validat automat, iar raportul afişează clar dacă sunt respectate toate structurile şi restricţiile impuse. După validare, orarul poate fi exportat direct în PDF şi Excel, gata de trimis către studenţi sau de tipărit

Slide 12— Comparaţie AI vs algoritm clasic

Aici am realizat o comparaţie directă între cele două metode implementate: soluţia cu AI, folosind GPT-4, şi algoritmul clasic Python.

Se observă că AI-ul oferă o flexibilitate mult mai mare – regulile pot fi adaptate imediat, fără a rescrie codul.

La personalizare şi interactivitate, AI-ul permite utilizatorului să seteze direct în interfaţă reguli şi constrângeri, pe când algoritmul clasic rămâne mult mai rigid.

În schimb, complexitatea implementării este mai mică pentru AI: practic totul se mută în prompt, iar codul e mult mai simplu.

În final, soluţia AI este generalizabilă – poate fi aplicată uşor şi pe structuri diferite, nu doar pe cele clasice licenţă-master.

Slide 13 — Rezultate obţinute

Ca rezultate, am reuşit să digitalizez complet procesul de planificare academică, eliminând necesitatea realizării manuale a orarului.

În plus, am obţinut o reducere semnificativă a timpului şi eliminarea erorilor umane, având şi o validare automată a structurii şi a restricţiilor academice.

Iar soluţia bazată pe AI s-a dovedit superioară din punct de vedere al flexibilităţii şi adaptabilităţii faţă de algoritmul clasic, cu export rapid în PDF şi Excel

Slide 14— Perspective de dezvoltare

Pe viitor, îmi propun să extind aplicaţia prin adăugarea unor profiluri dedicate pentru administratori, profesori şi studenţi, alături de notificări automate.

De asemenea, vreau să testez modele AI locale şi să adaug opţiunea de generare offline, astfel încât aplicaţia să fie complet independentă.

Iar pentru o adaptare cât mai largă, intenţionez să dezvolt o versiune mobilă pentru Android şi iOS şi să includ suport pentru orare de examene sau activităţi extracurriculare.

Slide 15 —

Vă mulţumesc mult pentru atenţie.