Slide 1 și 2

Lucrarea mea se intitulează «Aplicație pentru planificare inteligentă utilizând tehnici de AI» și a fost realizată în cadrul Facultății de Automatică, Calculatoare, Inginerie Electrică și Electronică, sub coordonarea doamnei S.I. dr. ing. Mihaela Țiplea.  
Am ales acest subiect pentru că planificarea academică este un proces important, dar și consumator de timp, iar integrarea inteligenței artificiale poate aduce îmbunătățiri majore.

Slide 3 — Scopul și obiectivele lucrării

Scopul principal a fost automatizarea procesului de generare a orarului universitar, pentru a reduce timpul și erorile umane.  
Am ales să integrez o soluție AI pe care am comparat-o cu un algoritm clasic, pentru a evalua beneficiile aduse.  
Rezultatul final a fost o aplicație web modernă și scalabilă, adaptată nevoilor universitare.

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.  
Aplicația propusă gestionează complet profesorii, sălile, grupele și regulile academice, respectând constrângeri concrete, cum ar fi cursurile comune pe an sau pauzele miercuri între 14 și 16.  
Orarul poate fi generat fie printr-o soluție AI, folosind GPT-4, fie cu un algoritm Python propriu, astfel încât să putem compara obiectiv flexibilitatea și eficiența celor două abordări.

Slide 5 — Tehnologii utilizate

Pentru dezvoltarea aplicației am folosit un stack tehnologic modern, împărțit în mai multe componente.  
Frontend-ul este realizat în React, împreună cu Bootstrap, ceea ce a permis crearea unei interfețe responsive, ușor de utilizat pe orice dispozitiv, cu componente precum tabele dinamice și formulare pentru introducerea profesorilor, sălilor și grupelor.

Backend-ul rulează în Flask pe Python. Aici am implementat logica aplicației sub forma unui API REST, care gestionează cererile venite din frontend și comunică cu baza de date.  
Toate datele legate de profesori, săli, grupe, reguli și orare sunt stocate într-o bază de date MySQL, structurată în mai multe tabele interconectate prin chei externe, astfel încât să respecte relațiile dintre entități.

Pentru generarea clasică a orarului, am folosit un algoritm Python propriu, care parcurge zilele și intervalele orare, verificând disponibilitatea sălilor și a profesorilor, și aplicând reguli academice, cum ar fi cursurile comune pe an, seminariile pe grupe și laboratoarele pe subgrupe, plus pauzele miercuri între 14 și 16.

Pe partea de AI, am folosit GPT-4 de la OpenAI, formulând prompturi detaliate care includ lista disciplinelor, profesorilor, sălilor și regulile academice, cerând modelului să genereze un JSON structurat cu orarul rezultat.

În final, aplicația permite exportul orarului generat direct în PDF și Excel, folosind ReportLab, PyPDF2 și xlsxwriter, pentru ca orarul să fie gata de distribuit sau tipărit.

Slide 6— Arhitectura și funcționarea sistemului

Aplicația a fost construită pe o arhitectură pe trei niveluri, cu React în interfață, Flask ca API în backend și MySQL pentru gestionarea datelor.  
Baza de date este structurată în tabele separate pentru profesori, săli, grupe și reguli academice, toate legate prin chei externe pentru a reflecta relațiile reale dintre entități.

Am implementat reguli academice concrete, cum sunt cursurile comune pe an, seminariile pe grupe și laboratoarele pe subgrupe, cu pauză miercuri între 14 și 16.  
Acestea sunt validate direct printr-o structură JSON, astfel încât să asigur corectitudinea orarului.

Fluxul principal pornește cu introducerea datelor în interfață, continuă cu generarea orarului — fie prin AI, folosind GPT-4, fie printr-un algoritm Python propriu care parcurge zilele și intervalele orare aplicând toate restricțiile — după care orarul trece printr-un proces de validare automată și este exportat în PDF sau Excel.

Astfel am implementat o metodologie hibridă, care îmi permite să compar flexibilitatea și eficiența între soluția AI și cea clasică, validând în același timp automat structura JSON și regulile academice.

Slide 7 — Comparație AI vs algoritm clasic

Am realizat o comparație directă între cele două metode implementate: soluția bazată pe AI, unde am folosit GPT-4, și un algoritm clasic propriu, scris în Python.  
Diferențele sunt vizibile pe mai multe aspecte.

La flexibilitate, soluția AI reușește să se adapteze instant la reguli noi doar prin modificarea promptului, fără a schimba codul.  
De exemplu, am putut adăuga rapid cerințe precum cursuri comune pe an sau pauză miercuri 14-16 direct în textul transmis către GPT-4.

În schimb, algoritmul clasic necesită modificări manuale în codul Python pentru fiecare regulă nouă, ceea ce îl face mai rigid.

La timpul de execuție, ambele metode au avut performanțe comparabile în generarea efectivă a orarului, însă la adaptabilitate, AI-ul a fost net superior, fiind scalabil și capabil să integreze rapid liste noi de profesori, săli și discipline.

În ceea ce privește mentenanța, soluția AI este mult mai simplu de întreținut: schimbările se fac în prompt, fără să fie nevoie să regândesc algoritmul.  
În schimb, algoritmul Python clasic necesită rescriere parțială sau completă la schimbarea regulilor academice.

Astfel, în context universitar, unde apar frecvent noi constrângeri, AI-ul s-a dovedit mult mai potrivit

Slide 8 — Rezultate obținute

Ca rezultate obținute, am reușit să digitalizez complet procesul de planificare academică, prin automatizarea generării orarului universitar.  
Am redus semnificativ timpul necesar comparativ cu realizarea manuală și am eliminat erorile umane, validând automat structura JSON și respectarea regulilor, cum ar fi cursurile comune pe an sau pauzele miercuri între 14 și 16.  
Soluția AI s-a dovedit net superioară în ceea ce privește flexibilitatea și adaptabilitatea față de algoritmul clasic, iar la final orarul poate fi exportat direct în PDF și Excel, gata de distribuit.

Slide 9 — Perspective de dezvoltare

Pentru viitor, am prevăzut extinderea aplicației cu mai multe direcții importante.  
În primul rând, doresc să adaug profiluri dedicate pentru administratori, profesori și studenți, fiecare cu funcționalități specifice, precum și un sistem de notificări automate atunci când apar modificări în orar.

Apoi, îmi propun să optimizez partea tehnică prin integrarea și testarea altor modele AI locale, pentru a reduce dependența de servicii comerciale, dar și să dezvolt un mod de generare offline.

În final, vreau să adaptez aplicația și pentru orare de examene sau activități extracurriculare și să creez o versiune mobilă compatibilă cu Android și iOS, astfel încât utilizatorii să aibă acces rapid direct de pe telefon.

Slide 10 — Concluzii și mulțumiri

În concluzie, proiectul meu a demonstrat cum putem folosi tehnologiile moderne și inteligența artificială pentru a transforma complet procesul de planificare academică.  
Acest citat surprinde foarte bine modul în care am lucrat la acest proiect: prin pași mici, consecvenți, am reușit să ating un rezultat valoros.  
Vă mulțumesc mult pentru atenție.