1. Scopul și obiectivele lucrării
2. Structura lucrării
3. Descrierea subiectului abordat
4. Metodologie și implementare
5. Rezultate experimentale și comparații
6. Concluzii și direcții viitoare

**📊 Slide 1 – Scopul și obiectivele lucrării**

**Bullets:**

* Automatizarea generării orarului universitar
* Reducerea timpului și erorilor
* Integrare AI vs algoritm clasic
* Digitalizare proces planificare

🎤 **Ce spui:**  
„Lucrarea are ca scop dezvoltarea unei soluții software care să automatizeze generarea orarului universitar, reducând semnificativ timpul și eliminând erorile specifice planificării manuale, prin integrarea tehnologiilor AI și comparația lor cu o metodă clasică.”

**🏗 Slide 2 – Structura lucrării**

**Bullets:**

* Studiu problemă & soluții existente
* Specificare cerințe
* Arhitectură & proiectare
* Implementare & testare
* Rezultate & concluzii

🎤 **Ce spui:**  
„Lucrarea este structurată pe cinci capitole, pornind de la analiza problemei și a soluțiilor existente, continuând cu definirea cerințelor, proiectarea și implementarea aplicației, iar în final evaluarea rezultatelor și formularea concluziilor.”

**🖥 Slide 3 – Descrierea subiectului abordat**

**Bullets:**

* Aplicație web pentru generare orar
* Gestionare date: profesori, săli, grupe, reguli
* Export PDF/Excel
* Interfață modernă, validări vizuale

🎤 **Ce spui:**  
„Am dezvoltat o aplicație web completă, care gestionează datele despre profesori, săli și grupe, aplică reguli complexe și permite generarea automată a orarului, cu export direct în PDF și Excel și o interfață modernă cu validări vizuale.”

**⚙ Slide 4 – Metodologie și implementare**

**Bullets scurte:**

* **Arhitectură pe 3 niveluri: React, Flask, MySQL**
* **Principiu: introducere date → generare → validare → export**
* **Flux date: frontend ↔ backend ↔ DB**
* **Generare orar: AI (GPT-4/Mistral) sau algoritm clasic**
* **Validare automată structură JSON & reguli academice**

**🎤 Ce poți spune aici:  
„Am implementat o arhitectură pe trei niveluri, cu React în frontend, Flask în backend și MySQL pentru stocare.  
Principiul de funcționare al aplicației este simplu: utilizatorul introduce datele (profesori, săli, grupe, reguli), acestea sunt procesate în backend fie de un model AI fie de un algoritm clasic propriu, apoi orarul generat este validat automat conform regulilor academice și poate fi exportat direct în PDF sau Excel.  
Astfel, fluxul datelor este clar: frontend-ul comunică cu backend-ul, care interacționează cu baza de date, asigurând un sistem modular și ușor de extins.”**

**📈 Slide 5 – Rezultate experimentale și comparații**

**Bullets:**

* Orare complete & valide
* Timp de generare redus
* AI = flexibil, adaptabil
* Clasic = rigid, modificabil greu

🎤 **Ce spui:**  
„Rezultatele obținute confirmă validitatea orarului generat, timpul de execuție a fost redus semnificativ, iar AI-ul a demonstrat o flexibilitate mult mai mare față de soluția clasică, fiind capabil să respecte ușor reguli noi.”

**🏆 Slide 6 – Concluzii și direcții viitoare**

**Bullets:**

* Digitalizare completă planificare
* Reducere efort manual
* Integrare ICS, extindere licee
* Noi restricții pe săli/profesori

🎤 **Ce spui:**  
„Concluzionând, aplicația oferă un pas important spre digitalizarea completă a proceselor academice, reducând munca manuală și riscul de erori. Pe viitor, poate fi extinsă cu integrare directă în calendare, suport pentru alte tipuri de instituții și reguli avansate.”