LOGARG- LOGICAL ARGUMENTATION INTERFACE

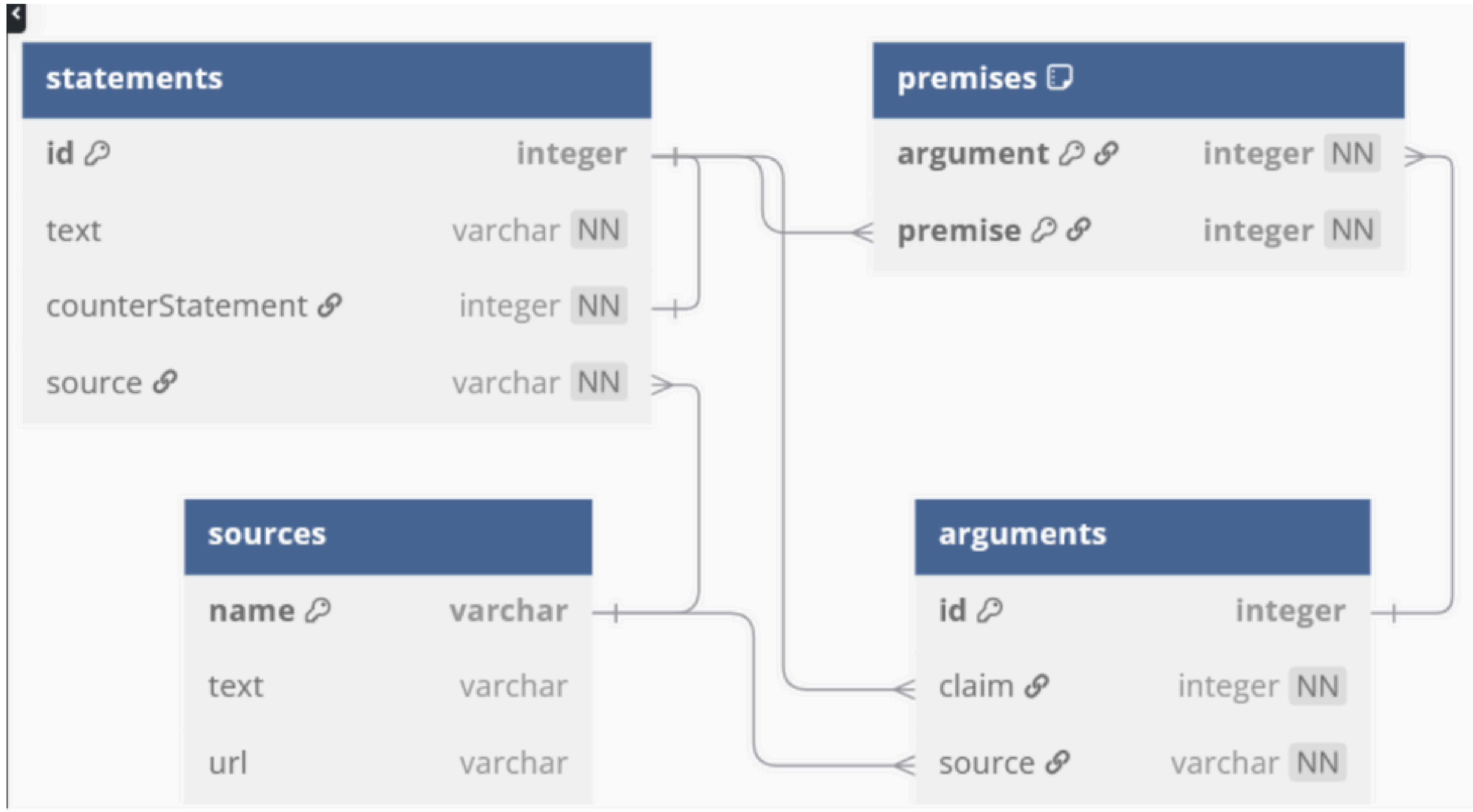
Frontend development- Final Presentation

Presented by: Sudarshan Gopalakrishna
M.Sc. Computer Science for Digital Media
Bauhaus-Universität Weimar

PROJECT OVERVIEW & OBJECTIVES

About the Project	Project Objective
LOGARG is a research-based platform for exploring logical argumentation.	Visualize argument structures from structured data.
It connects MySQL → Spring Boot → Angular for structured debates.	Design a summary-based exploration mode for reflective interaction
This semester focused on frontend design and interactive experience.	Document APIs aligned with the database schema for smooth integration.

DATABASE DESIGN & BACKEND INTEGRATION

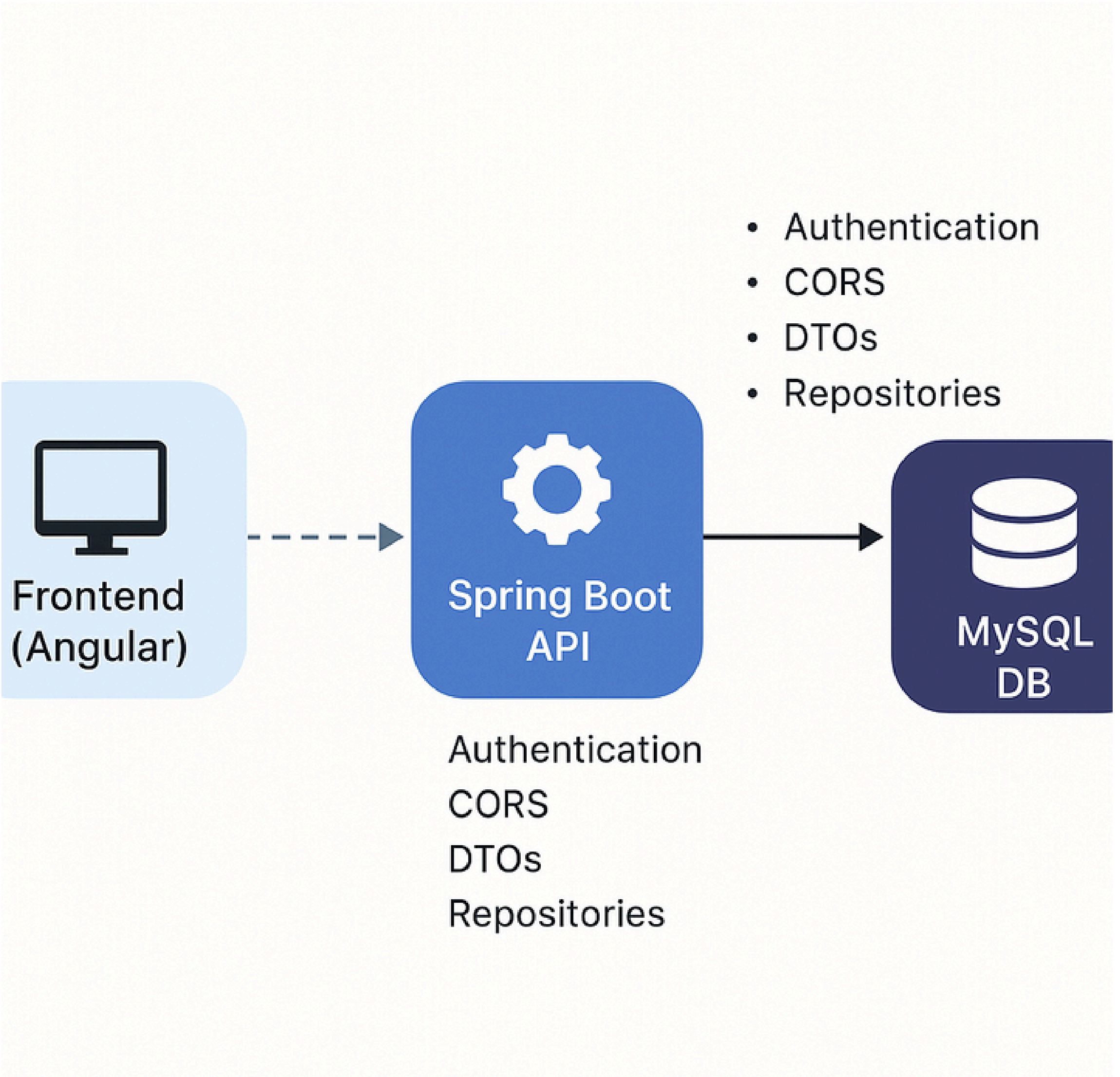


statements		
id 🔑	integer	
text	varchar	NN
counterStatement 🔗	integer	NN
source 🔗	varchar	NN

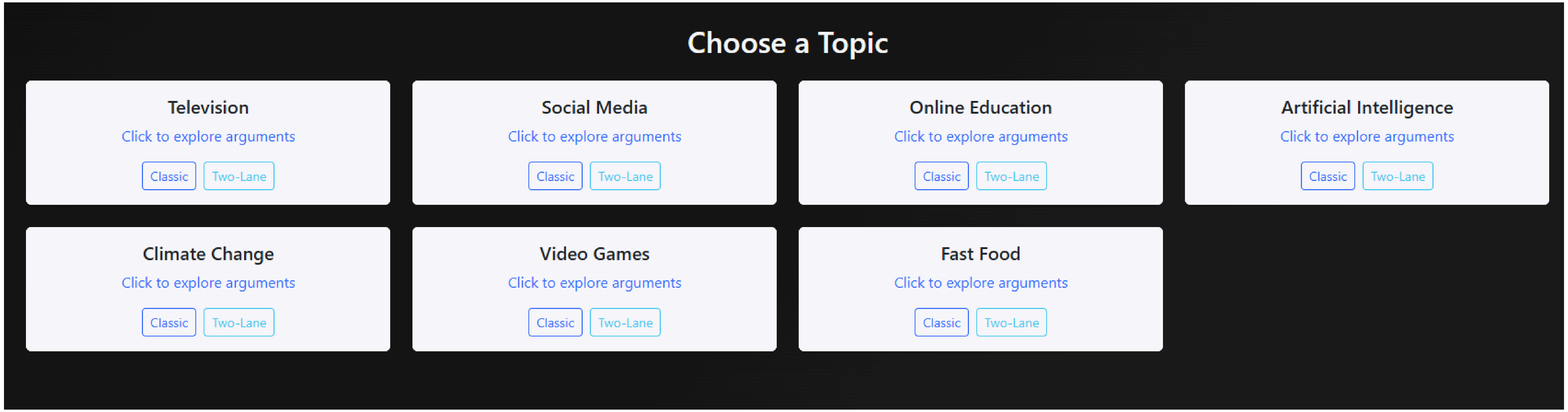
premises		
argument 🔑 🔗	integer	NN
premise 🔑 🔗	integer	NN

sources	
name 🔑	varchar
text	varchar
url	varchar

arguments		
id 🔑	integer	
claim 🔗	integer	NN
source 🔗	varchar	NN



HOMEPAGE OVERVIEW



Choose a Topic

Television
Click to explore arguments
Classic Two-Lane

Social Media
Click to explore arguments
Classic Two-Lane

Online Education
Click to explore arguments
Classic Two-Lane

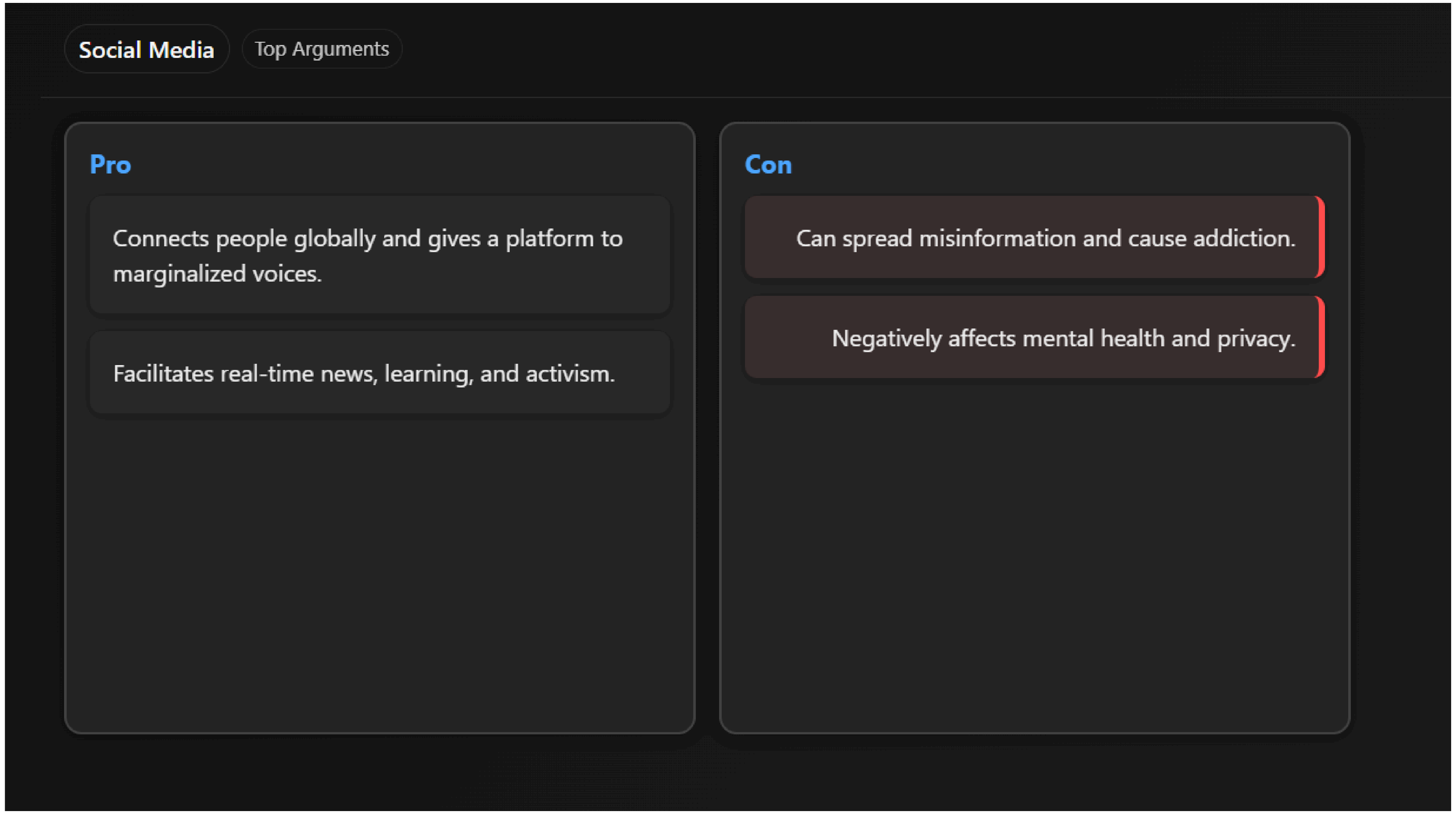
Artificial Intelligence
Click to explore arguments
Classic Two-Lane

Climate Change
Click to explore arguments
Classic Two-Lane

Video Games
Click to explore arguments
Classic Two-Lane

Fast Food
Click to explore arguments
Classic Two-Lane

TASK 1: VISUALIZE CLASSIC ARGUMENT STRUCTURES (ARGS.ME)



Pro

Connects people globally and gives a platform to marginalized voices.

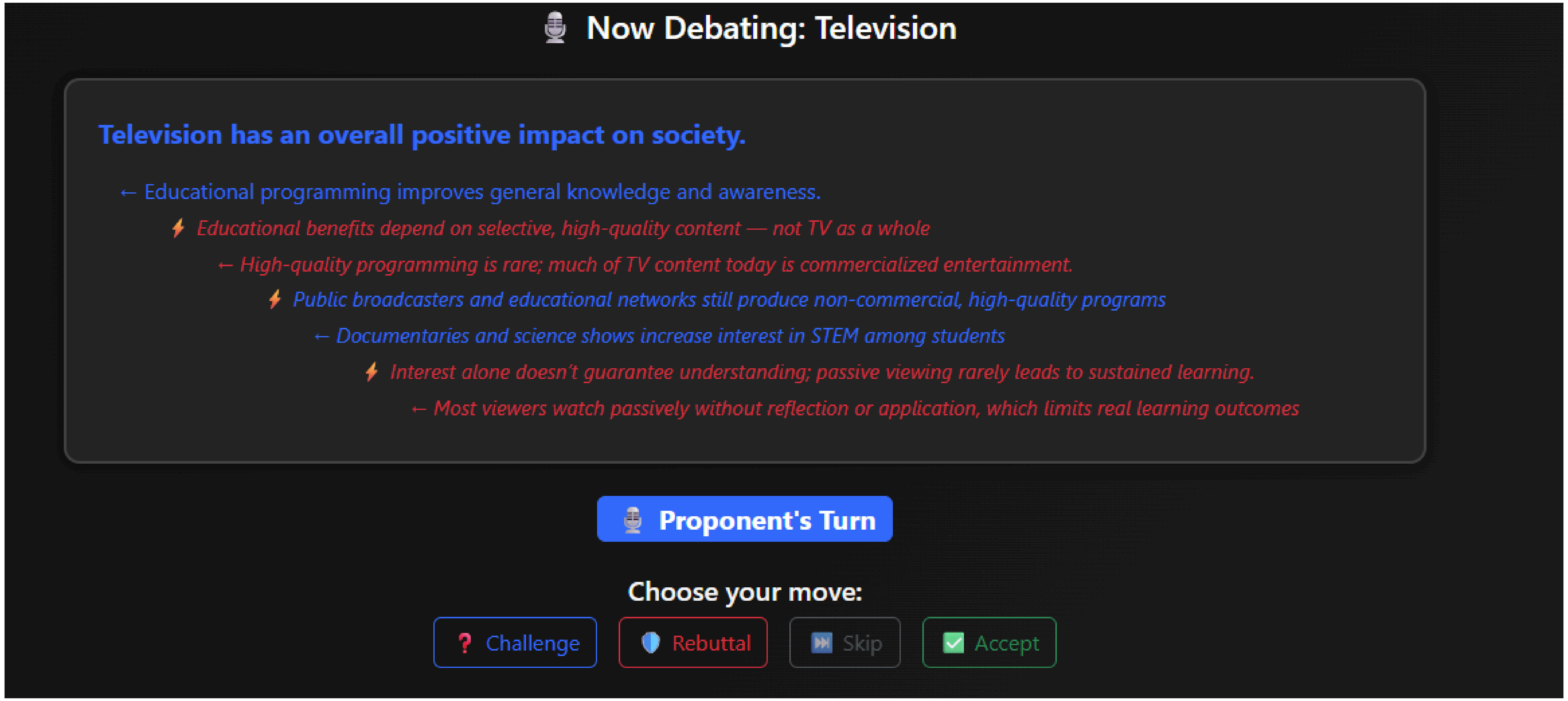
Facilitates real-time news, learning, and activism.

Con

Can spread misinformation and cause addiction.

Negatively affects mental health and privacy.

TASK 2: CLASSIC VIEW (INTERACTIVE EXPLORATION MODE)



🎙 Now Debating: Television

Television has an overall positive impact on society.

← Educational programming improves general knowledge and awareness.

⚡ *Educational benefits depend on selective, high-quality content — not TV as a whole*

← *High-quality programming is rare; much of TV content today is commercialized entertainment.*

⚡ *Public broadcasters and educational networks still produce non-commercial, high-quality programs*

← *Documentaries and science shows increase interest in STEM among students*

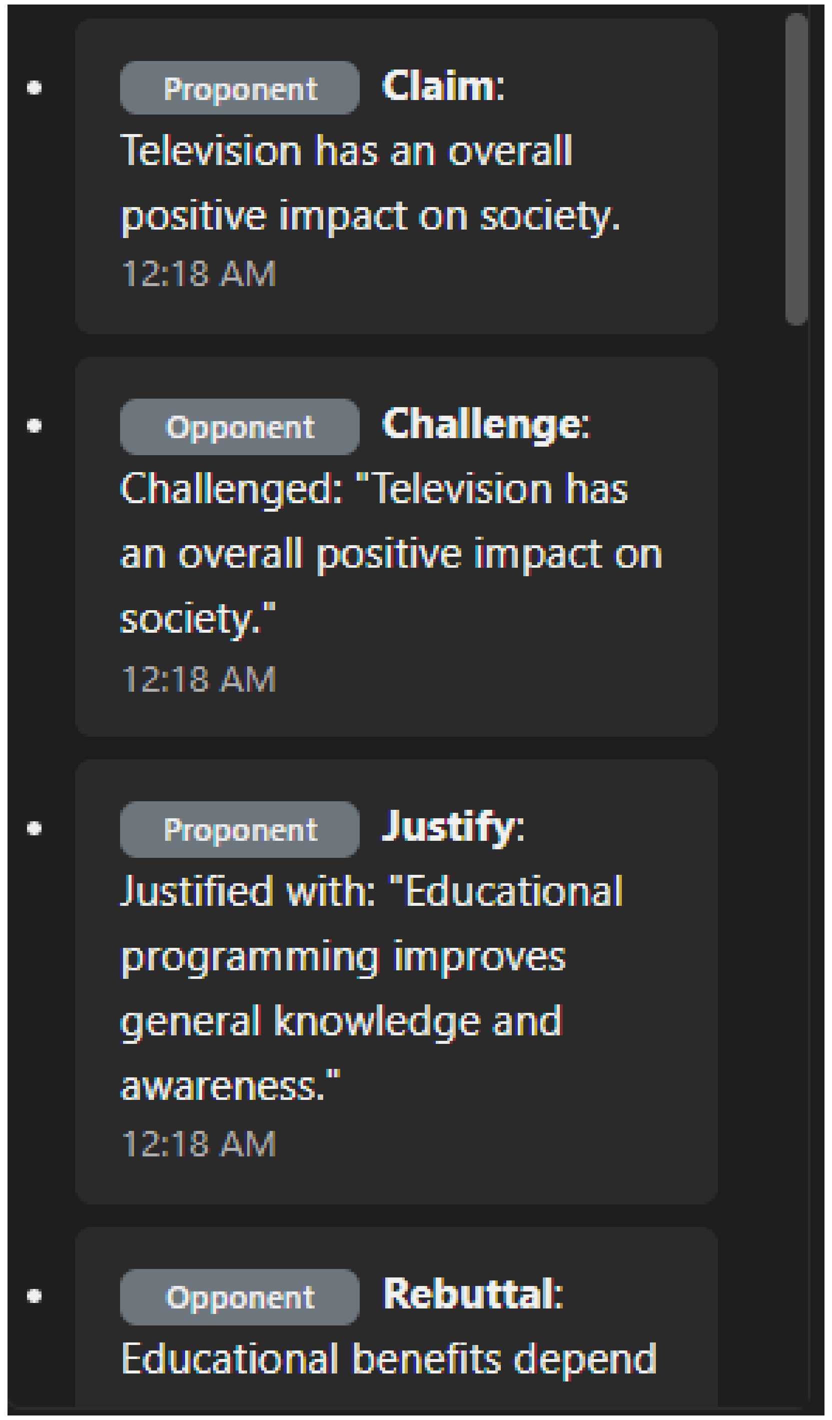
⚡ *Interest alone doesn't guarantee understanding; passive viewing rarely leads to sustained learning.*

← *Most viewers watch passively without reflection or application, which limits real learning outcomes*

ALLOWED MOVES & INTERACTION RULES

1. Opponent's First Turn
   - Can only Challenge or Accept the opening claim.
2. When a Justification Is Waiting for Response
   - The other side can Challenge, Rebut, or Accept it.
   - Gives freedom to question or agree with a reason.
3. Proponent Responding to Opponent's Rebuttal
   - Can only Accept the rebuttal or Challenge it.
   - Prevents infinite back-and-forth rebuttals.
4. Duplicate Challenges / Rebuttals
   - Automatically blocked by the component logic.
   - Avoids repeating the same move on one argument.
5. Default Case
   - If none of the above applies, all moves are available:
   - Challenge, Rebut, Accept, Skip

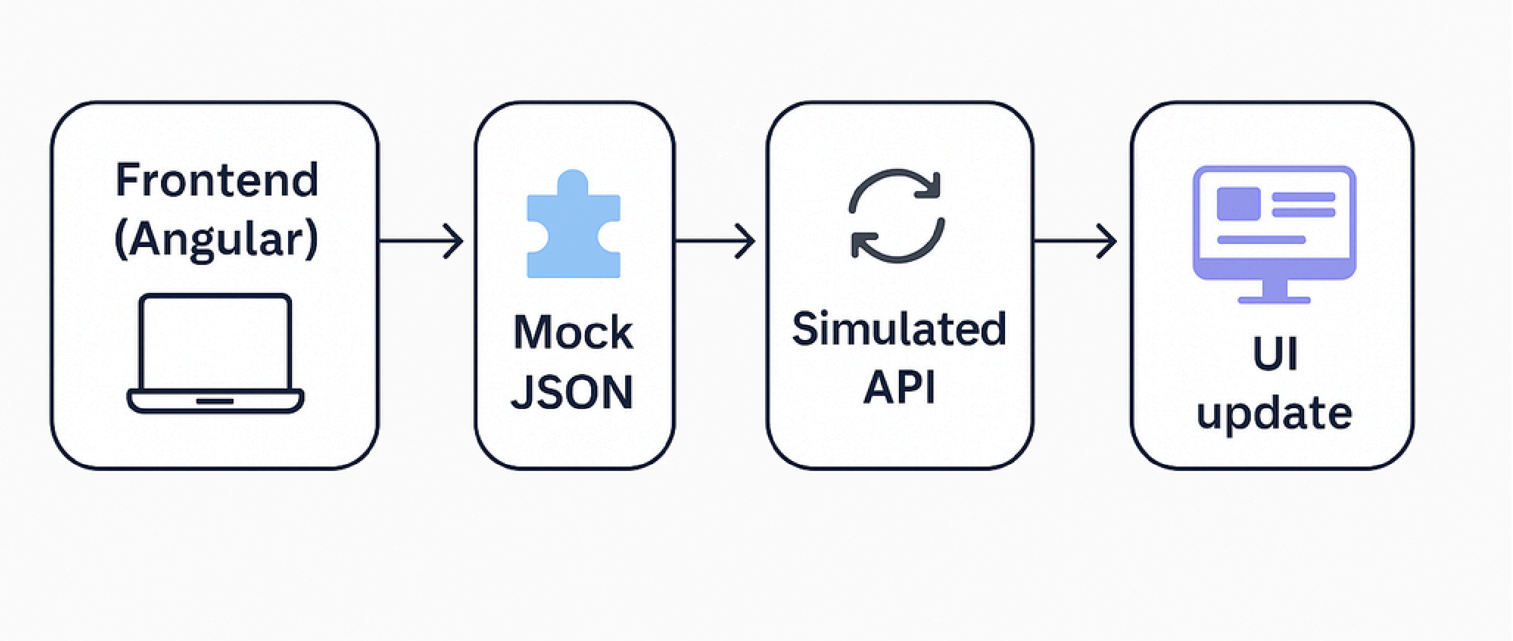
MOVE HISTORY: TRACKING THE DEBATE FLOW



- **Claim**:
  Television has an overall positive impact on society.
  12:18 AM
- **Challenge**:
  Challenged: "Television has an overall positive impact on society."
  12:18 AM
- **Justify**:
  Justified with: "Educational programming improves general knowledge and awareness."
  12:18 AM
- **Rebuttal**:
  Educational benefits depend

TASK 3: SIMULATING BACKEND CALLS (MOCK API)

- Implemented mock API endpoints to simulate real backend behavior.
- Used local JSON files for topics, arguments, and rebuttals.
- Enabled frontend testing without needing Spring Boot running.
- Ensured data fetching, filtering, and dynamic loading worked correctly.



TASK 4: FUTURE IMPROVEMENTS (LOGIC & UI/UX)

1. **Smarter Move Logic**
   - Make the Allowed Moves system more flexible and context-aware.
   - Add new rule conditions for deeper debate states and exceptions.
   - Improve clarity when certain moves are blocked or triggered.
2. **Interactive Argument Tree**
   - Replace indentation with a visual debate graph (e.g., D3.js / ngx-graph).
   - Allow expand/collapse of justifications and rebuttals. (Attempted ,but it broke the exisiting code!)
   - Use Argsme Component as a single component to see both
3. **Better Move Guidance**
   - Show tooltips or highlights for valid “Challenge” / “Rebut” targets.
   - Add small animations or icons for accepted/rejected moves.
4. **Enhanced Layout & Readability**
   - Two-lane layout with sticky Proponent/Opponent headers.
   - Collapsible sections for long texts and balanced spacing.

DOCUMENTATION

- All code files are well-commented, with clear descriptions for every property, variable, and function.
- Includes detailed notes on database configuration and environment setup.
- Contains README files for frontend and backend with setup steps and explanations.
- Provides API documentation outlining all endpoints and their usage.
- Includes a short section on future improvements for logic and UI/UX.

THANK YOU