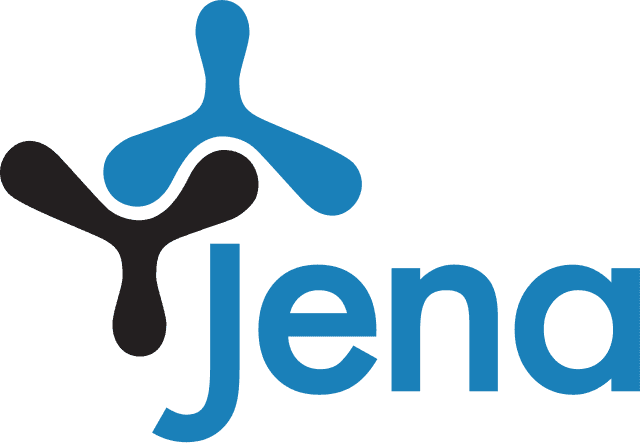
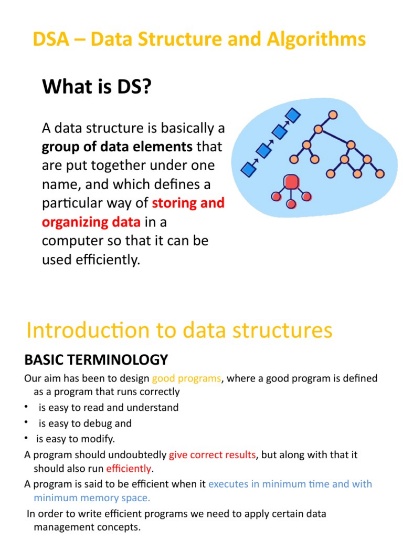
**Pixella- AI Enabled Education Mentor**

Pixella will be a web-application which will be the new EdTutor. It will allow to learn on specific topics which will be stored in the database(graphs) and get a visualized experience for enhanced learning.

**Design and workflow:**

* Extracts information from textbooks and research papers, transforms the data into structured semantic relationships using RDF, and stores the knowledge in a graph database for efficient querying and learning. The system leverages **Apache Jena** for RDF model creation and **Neo4j** graph storage. I have opted Apache Jena as custom ETL pipelines are very lengthy and code optimization is better in Apache Jena.



**Python Parsing**

Data Sources

**RDF Triplets**

**QUERYING**



**AWS S3**

**BERT TRANSFROMER**

**REACT**

**MANIM API**

**GET,POST**

NODES AND GRAPHS

**FASTAPI**

* Once the Data has been stored now OpenAI vector Embeddings will be applied using sentence transformer for semantic search. I will make a customized chat prompt template with a custom code like

You are an educational content creator. Given the following concept details extracted from books:

{concept\_text}

Generate:

1. A slide title summarizing the concept.

2. A detailed script elaborating on each bullet point.

3. Suggestions for animations or diagrams to visualize the concept.

Format your response as JSON with keys: title, bullets, script, animations.

"""

* **AI Content Generation Using Langchain**.

Use Langchain to make a prompt as mentioned above and ma e a proper chain of code. We will use a SequentialChain to first generate slides, the scripts and then animation transcripts.

Use OPENAI LLM and fine tune using PEFT parameter LoRa to get accurate results.

Use langchain ConersationChain or Memory to remember the inputs.

* **Manim Automation:**

Use manim Python API to create the scenes, automnate creation of text,bullet points and more.

The Output will be rendered video files(MP4)

* **FRONTEND**

The frontend will be connected using FASTAPI to get the GET,POST api to the backend and make a react application to connect with it>

Pseudo Code:

from fastapi import FastAPI

from pydantic import BaseModel

app = FastAPI()

class QueryRequest(BaseModel):

query: str

@app.post("/api/generate-video")

async def generate\_video(request: QueryRequest):

# 1. Retrieve concept from knowledge graph

# 2. Generate slides, script, animation

# 3. Return structured response with URLs

response = {

"slides": [

{"title": "Graph Theory", "bullets": ["Graphs model relations", "Vertices and edges"]},

# more slides...

],

"script": "Graph theory studies graphs, which are mathematical structures...",

"video\_url": "https://your-storage.com/generated\_video.mp4"

}

return response

* **STORAGE DATABASES**

AWS S3 buckets will be used for the cloud storage as it is one of the best services available

FRONTEND PROTOTYPE:



MP4 Video URL

SERACH BAR/INPUT