Engr 691: Deep Learning
Project 5 Report: RAG-enhanced Course Recommender

Abdul-Rashid Zakaria 10992191

May 2025

1 Introduction

This report summarizes the course recommender implemented using the DeepSeek-R1-Distill-Llama-70B model. It is divided into sections that provide the key details of the project's architecture, components, and submodules that produce tailored course recommendations.

2 System Architecture and Components

2.1 User Interface Layer

The system currently uses a command-line interface that collects essential student information, however, the backend has also been implemented to support a much more user-friendly input that is a front-end built on react is also being worked on at the time of producing this report.

The essential information collection includes:

- Prior courses (entered as comma-separated strings)
- Current academic department of the user
- Degree level (with validation for "undergraduate" or "graduate")
- Target course (the course the student ultimately wants to take)

2.2 Data Models

Two primary data models help collect the data into a structured format.

- Student Model: Encapsulates academic background information, including prior courses, department, and degree level
- Course Database: Stores course information, prerequisites, and relationships between courses

2.3 Learning Plan Generator

The core of the system is based on the Retrieval-augmented generation (RAG) and allows a multi-turn interaction approach.

- Initializes connection to a language model service using environment variables
- Processes student information and target course data
- Delivers a comprehensive learning plan

2.4 External Knowledge Integration

The system incorporates an engineering course catalog (PDF) as an external knowledge source. The catalog was created by scrapping all the course catalogs on the Olemiss website using a modified scrapper built on the work of William Panlener, github. However, only engineering courses were used for demonstration in this project. In addition, to simplify it further, we saved the catalog as a pdf which is analyzed. A better implementation would involve fine-tuning our LLM on all the course catalogs.

- Loads and parses catalog text using PyMuPDF (fitz library)
- Provides ground truth for course existence and prerequisites
- Allows verification against actual course offerings

2.5 RAG Methodology

The system implements RAG, a methodology that significantly improves recommendation quality through several key mechanisms.

The system extracts text from an engineering catalog PDF and makes it searchable through the implemented search method. The system also maintains context windows (500 characters by default) around matching terms. Automatically adds the target course as a key search term. It uses a predefined set of educational topics, such as calculus, programming, and statistics. The system employs regular expressions to identify course codes in various formats, such as "CSCI 101" and "MATH240." It also combines adjacent words when detecting course names and numbers. To prevent prompt overflow and maintain focus, each term will retrieve a maximum of 2 snippets. A total limit of 8 snippets is enforced, and text is truncated to a maximum of 4000 characters.

2.6 Multi-Turn Interaction Process

The system breaks down the complex task of learning plan generation into four sequential steps, with each step building upon the previous output.

- Step 1: Knowledge Assessment

  It assesses the student's existing knowledge based on their background. It emphasizes mathematical foundations, programming skills, domain knowledge, and related concepts and provides a structured evaluation of current understanding.

- Step 2: Gap analysis

  Identifies specific knowledge and skill gaps based on the assessment from Step 1. It considers mathematical prerequisites, programming skills, theoretical foundations, and practical experience, generating a clear list of gaps that need to be addressed.

- Step 3: Course selection with RAG

  Conducts a catalog search using terms derived from the gap analysis. Enhances the prompt with relevant catalog information. Recommends specific courses that address the identified gaps. Verifies the availability of each course in the current catalog. Justifies recommendations based on the student's background. Indicates whether each course is essential or optional and notes any prerequisites for the recommended courses.

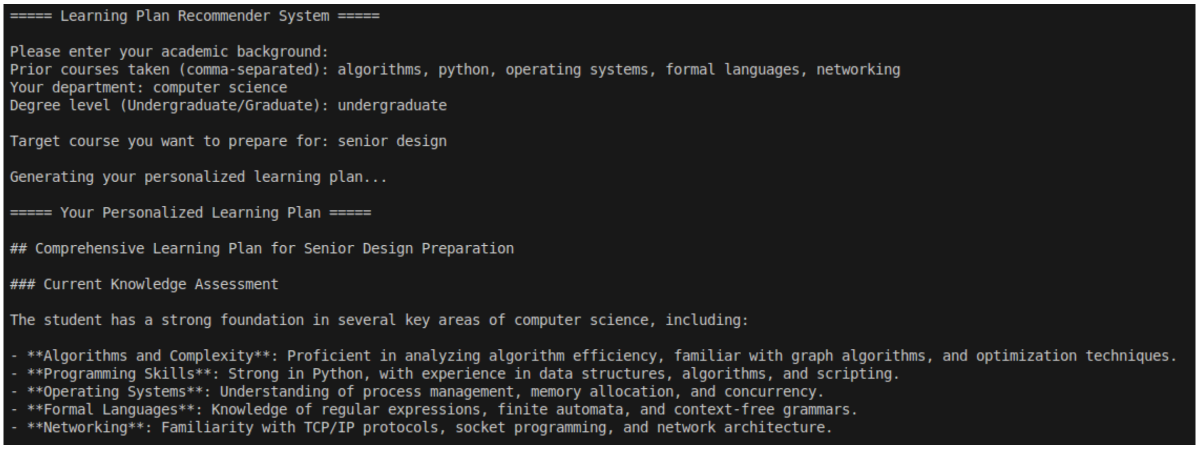
- Step 4: Final plan generation

  Consolidates all previous insights into a detailed learning plan. Summarizes existing knowledge and identifies gaps. Offers a sequential outline of courses and activities. Suggests additional materials and resources. Recommends a feasible timeline for completion.

In conclusion, user input is collected through the command-line interface and structured into a Student object. The system initializes and connects to the R1 LLM service, the system then orchestrates a four-step process where each step builds on the previous step's output. The RAG mechanism enriches recommendations with actual catalog data The final learning plan is presented to the user with actionable recommendations.

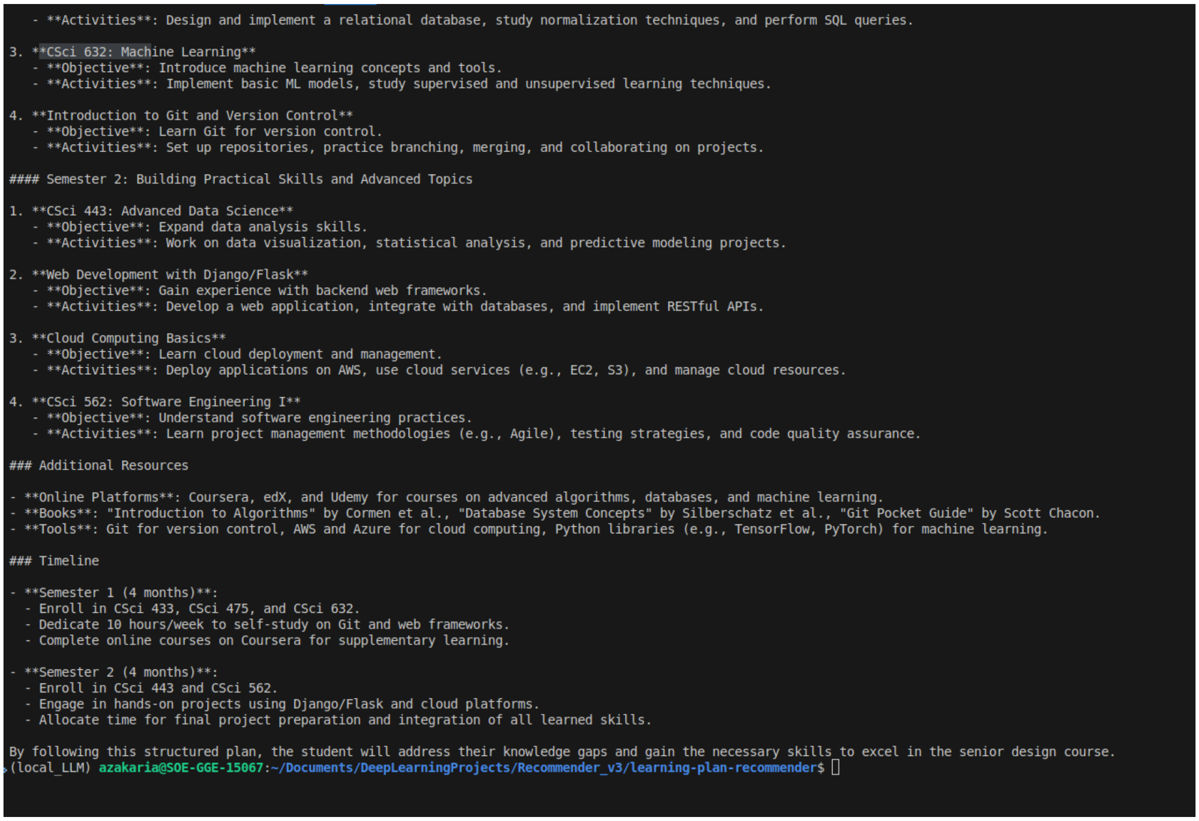
3 Sample Run of the System

- Profile 1: Undergraduate student in Computer Science targeting to take the senior design course.



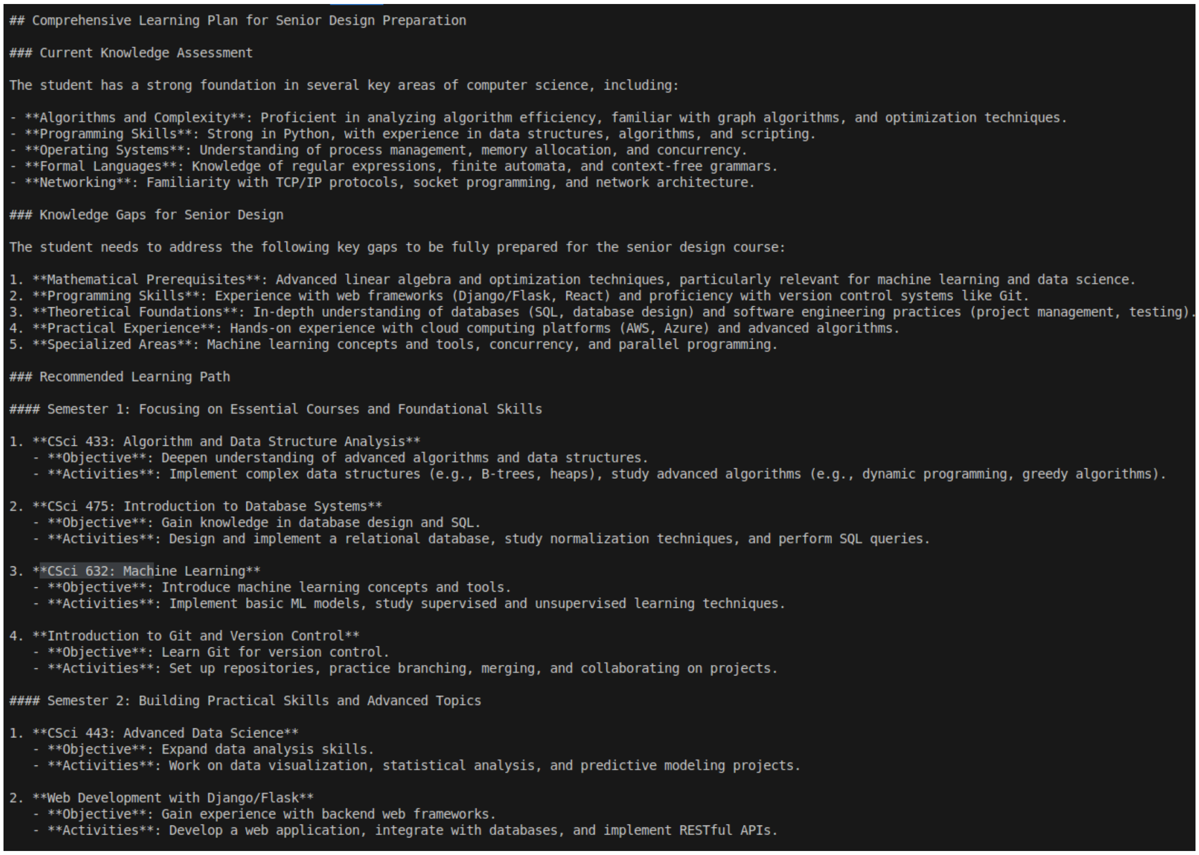
```
===== Learning Plan Recommender System =====

Please enter your academic background:
Prior courses taken (comma-separated): algorithms, python, operating systems, formal languages, networking
Your department: computer science
Degree level (Undergraduate/Graduate): undergraduate

Target course you want to prepare for: senior design

Generating your personalized learning plan...

===== Your Personalized Learning Plan =====

## Comprehensive Learning Plan for Senior Design Preparation

### Current Knowledge Assessment

The student has a strong foundation in several key areas of computer science, including:

- **Algorithms and Complexity**: Proficient in analyzing algorithm efficiency, familiar with graph algorithms, and optimization techniques.
- **Programming Skills**: Strong in Python, with experience in data structures, algorithms, and scripting.
- **Operating Systems**: Understanding of process management, memory allocation, and concurrency.
- **Formal Languages**: Knowledge of regular expressions, finite automata, and context-free grammars.
- **Networking**: Familiarity with TCP/IP protocols, socket programming, and network architecture.
```

Figure 1: Essential information of the student.



```
   - **Activities**: Design and implement a relational database, study normalization techniques, and perform SQL queries.

3. **CSci 632: Machine Learning**
   - **Objective**: Introduce machine learning concepts and tools.
   - **Activities**: Implement basic ML models, study supervised and unsupervised learning techniques.

4. **Introduction to Git and Version Control**
   - **Objective**: Learn Git for version control.
   - **Activities**: Set up repositories, practice branching, merging, and collaborating on projects.

#### Semester 2: Building Practical Skills and Advanced Topics

1. **CSci 443: Advanced Data Science**
   - **Objective**: Expand data analysis skills.
   - **Activities**: Work on data visualization, statistical analysis, and predictive modeling projects.

2. **Web Development with Django/Flask**
   - **Objective**: Gain experience with backend web frameworks.
   - **Activities**: Develop a web application, integrate with databases, and implement RESTful APIs.

3. **Cloud Computing Basics**
   - **Objective**: Learn cloud deployment and management.
   - **Activities**: Deploy applications on AWS, use cloud services (e.g., EC2, S3), and manage cloud resources.

4. **CSci 562: Software Engineering I**
   - **Objective**: Understand software engineering practices.
   - **Activities**: Learn project management methodologies (e.g., Agile), testing strategies, and code quality assurance.

### Additional Resources

- **Online Platforms**: Coursera, edX, and Udemy for courses on advanced algorithms, databases, and machine learning.
- **Books**: "Introduction to Algorithms" by Cormen et al., "Database System Concepts" by Silberschatz et al., "Git Pocket Guide" by Scott Chacon.
- **Tools**: Git for version control, AWS and Azure for cloud computing, Python libraries (e.g., TensorFlow, PyTorch) for machine learning.

### Timeline

- **Semester 1 (4 months)**:
  - Enroll in CSci 433, CSci 475, and CSci 632.
  - Dedicate 10 hours/week to self-study on Git and web frameworks.
  - Complete online courses on Coursera for supplementary learning.

- **Semester 2 (4 months)**:
  - Enroll in CSci 443 and CSci 562.
  - Engage in hands-on projects using Django/Flask and cloud platforms.
  - Allocate time for final project preparation and integration of all learned skills.

By following this structured plan, the student will address their knowledge gaps and gain the necessary skills to excel in the senior design course.
(local_LLM) azakaria@SOE-GGE-15067:~/Documents/DeepLearningProjects/Recommender_v3/learning-plan-recommender$ 
```

Figure 3: Final course recommendation and timeline for the student.

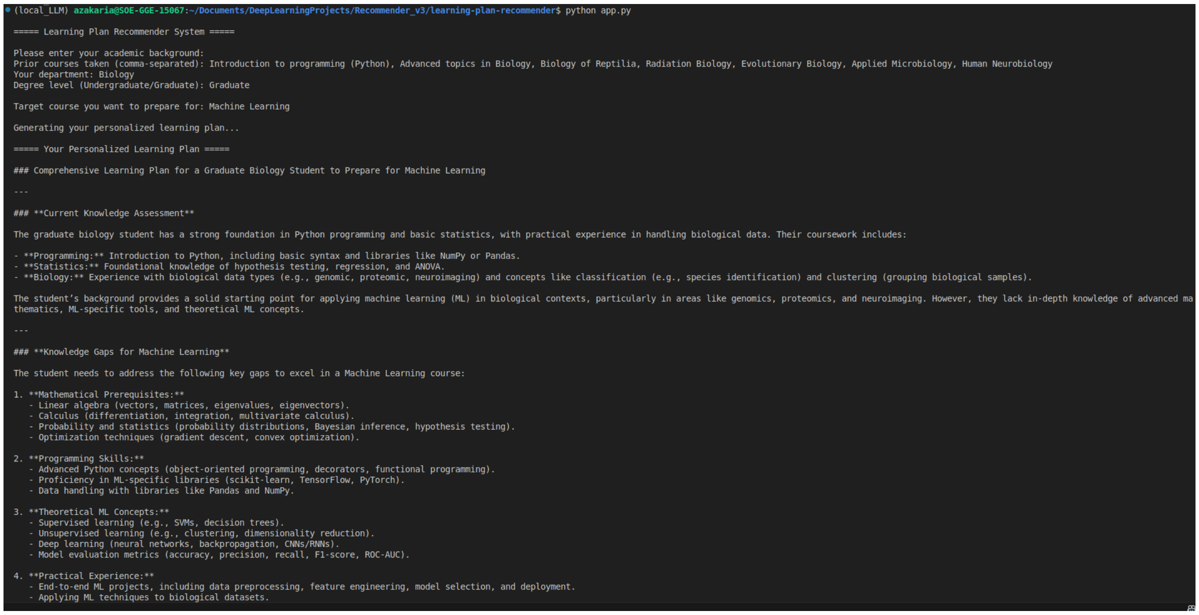


```
## Comprehensive Learning Plan for Senior Design Preparation

### Current Knowledge Assessment

The student has a strong foundation in several key areas of computer science, including:

- **Algorithms and Complexity**: Proficient in analyzing algorithm efficiency, familiar with graph algorithms, and optimization techniques.
- **Programming Skills**: Strong in Python, with experience in data structures, algorithms, and scripting.
- **Operating Systems**: Understanding of process management, memory allocation, and concurrency.
- **Formal Languages**: Knowledge of regular expressions, finite automata, and context-free grammars.
- **Networking**: Familiarity with TCP/IP protocols, socket programming, and network architecture.

### Knowledge Gaps for Senior Design

The student needs to address the following key gaps to be fully prepared for the senior design course:

1. **Mathematical Prerequisites**: Advanced linear algebra and optimization techniques, particularly relevant for machine learning and data science.
2. **Programming Skills**: Experience with web frameworks (Django/Flask, React) and proficiency with version control systems like Git.
3. **Theoretical Foundations**: In-depth understanding of databases (SQL, database design) and software engineering practices (project management, testing).
4. **Practical Experience**: Hands-on experience with cloud computing platforms (AWS, Azure) and advanced algorithms.
5. **Specialized Areas**: Machine learning concepts and tools, concurrency, and parallel programming.

### Recommended Learning Path

#### Semester 1: Focusing on Essential Courses and Foundational Skills

1. **CSci 433: Algorithm and Data Structure Analysis**
   - **Objective**: Deepen understanding of advanced algorithms and data structures.
   - **Activities**: Implement complex data structures (e.g., B-trees, heaps), study advanced algorithms (e.g., dynamic programming, greedy algorithms).

2. **CSci 475: Introduction to Database Systems**
   - **Objective**: Gain knowledge in database design and SQL.
   - **Activities**: Design and implement a relational database, study normalization techniques, and perform SQL queries.

3. **CSci 632: Machine Learning**
   - **Objective**: Introduce machine learning concepts and tools.
   - **Activities**: Implement basic ML models, study supervised and unsupervised learning techniques.

4. **Introduction to Git and Version Control**
   - **Objective**: Learn Git for version control.
   - **Activities**: Set up repositories, practice branching, merging, and collaborating on projects.

#### Semester 2: Building Practical Skills and Advanced Topics

1. **CSci 443: Advanced Data Science**
   - **Objective**: Expand data analysis skills.
   - **Activities**: Work on data visualization, statistical analysis, and predictive modeling projects.

2. **Web Development with Django/Flask**
   - **Objective**: Gain experience with backend web frameworks.
   - **Activities**: Develop a web application, integrate with databases, and implement RESTful APIs.
```

Figure 2: Knowledge Assessment of the student.

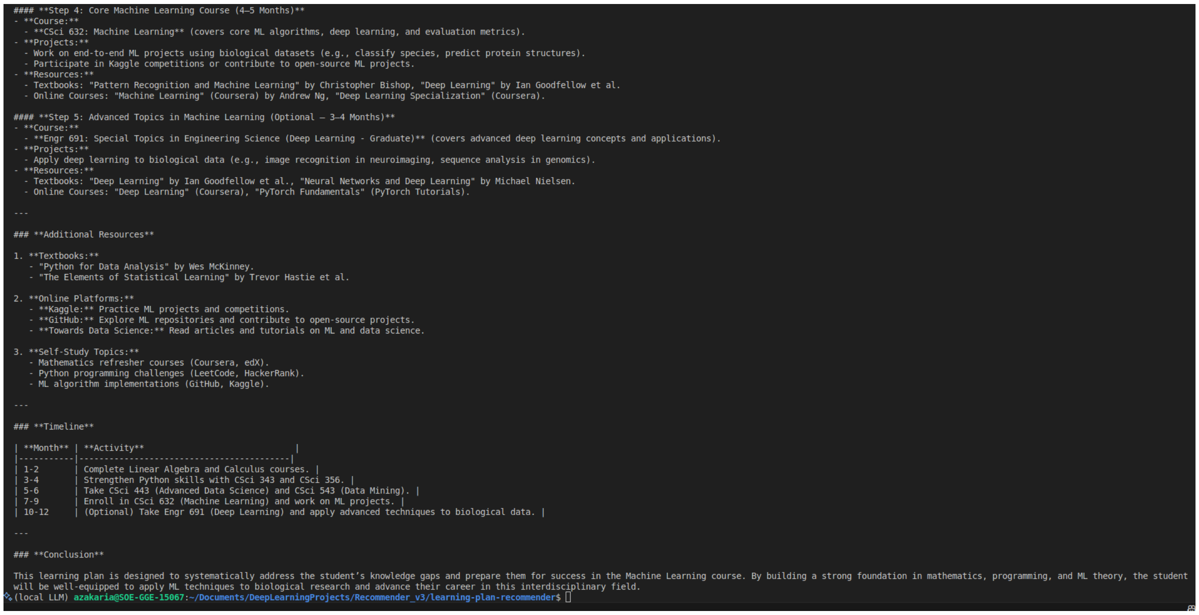
The system can make reasonable recommendations for the student, including self-study, to fill any knowledge gaps that may not be resolved by taking classes.

- Profile 2: Graduate student in Biology targeting to take the machine learning course.



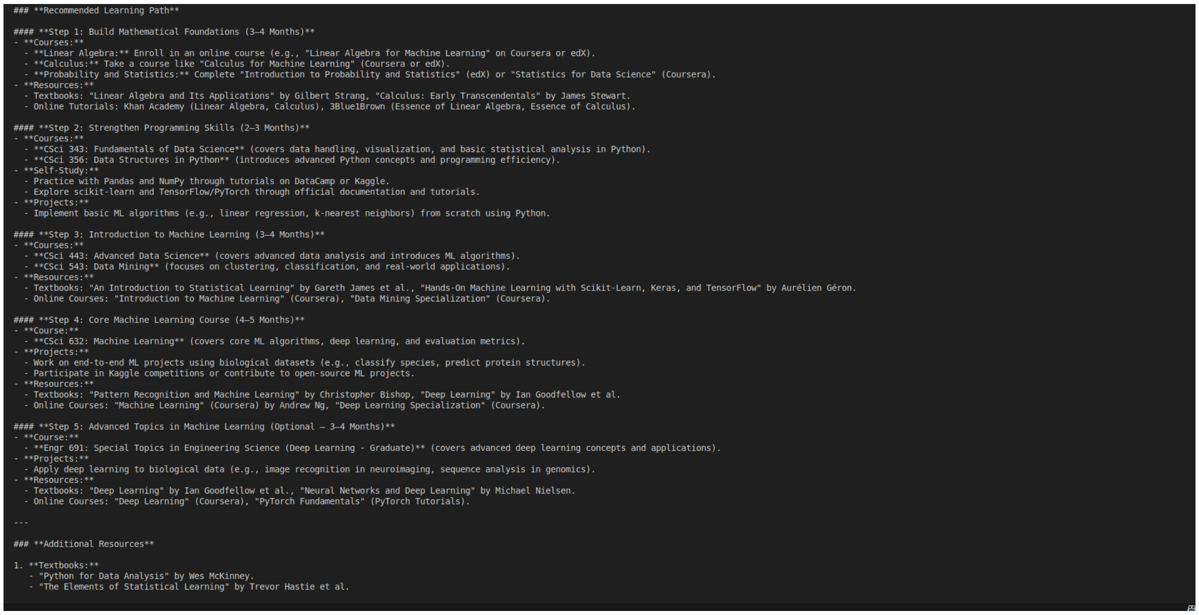
```
(local_LLM) azakaria@SOE-GGE-15067:~/Documents/DeepLearningProjects/Recommender_v3/learning-plan-recommender$ python app.py

===== Learning Plan Recommender System =====

Please enter your academic background:
Prior courses taken (comma-separated): Introduction to programming (Python), Advanced topics in Biology, Biology of Reptilia, Radiation Biology, Evolutionary Biology, Applied Microbiology, Human Neurobiology
Your department: Biology
Degree level (Undergraduate/Graduate): Graduate

Target course you want to prepare for: Machine Learning

Generating your personalized learning plan...

===== Your Personalized Learning Plan =====

### Comprehensive Learning Plan for a Graduate Biology Student to Prepare for Machine Learning

---

### **Current Knowledge Assessment**

The graduate biology student has a strong foundation in Python programming and basic statistics, with practical experience in handling biological data. Their coursework includes:

- **Programming:** Introduction to Python, including basic syntax and libraries like NumPy or Pandas.
- **Statistics:** Foundational knowledge of hypothesis testing, regression, and ANOVA.
- **Biology:** Experience with biological data types (e.g., genomic, proteomic, neuroimaging) and concepts like classification (e.g., species identification) and clustering (grouping biological samples).

The student’s background provides a solid starting point for applying machine learning (ML) in biological contexts, particularly in areas like genomics, proteomics, and neuroimaging. However, they lack in-depth knowledge of advanced ma
thematics, ML-specific tools, and theoretical ML concepts.

---

### **Knowledge Gaps for Machine Learning**

The student needs to address the following key gaps to excel in a Machine Learning course:

1. **Mathematical Prerequisites:**
   - Linear algebra (vectors, matrices, eigenvalues, eigenvectors).
   - Calculus (differentiation, integration, multivariate calculus).
   - Probability and statistics (probability distributions, Bayesian inference, hypothesis testing).
   - Optimization techniques (gradient descent, convex optimization).

2. **Programming Skills:**
   - Advanced Python concepts (object-oriented programming, decorators, functional programming).
   - Proficiency in ML-specific libraries (scikit-learn, TensorFlow, PyTorch).
   - Data handling with libraries like Pandas and NumPy.

3. **Theoretical ML Concepts:**
   - Supervised learning (e.g., SVMs, decision trees).
   - Unsupervised learning (e.g., clustering, dimensionality reduction).
   - Deep learning (neural networks, backpropagation, CNNs/RNNs).
   - Model evaluation metrics (accuracy, precision, recall, F1-score, ROC-AUC).

4. **Practical Experience:**
   - End-to-end ML projects, including data preprocessing, feature engineering, model selection, and deployment.
   - Applying ML techniques to biological datasets.
```

Figure 4: Essential information of the student.



```
#### **Step 4: Core Machine Learning Course (4–5 Months)**
- **Course:**
  - **CSci 632: Machine Learning** (covers core ML algorithms, deep learning, and evaluation metrics).
- **Projects:**
  - Work on end-to-end ML projects using biological datasets (e.g., classify species, predict protein structures).
  - Participate in Kaggle competitions or contribute to open-source ML projects.
- **Resources:**
  - Textbooks: "Pattern Recognition and Machine Learning" by Christopher Bishop, "Deep Learning" by Ian Goodfellow et al.
  - Online Courses: "Machine Learning" (Coursera) by Andrew Ng, "Deep Learning Specialization" (Coursera).

#### **Step 5: Advanced Topics in Machine Learning (Optional – 3–4 Months)**
- **Course:**
  - **Engr 691: Special Topics in Engineering Science (Deep Learning - Graduate)** (covers advanced deep learning concepts and applications).
- **Projects:**
  - Apply deep learning to biological data (e.g., image recognition in neuroimaging, sequence analysis in genomics).
- **Resources:**
  - Textbooks: "Deep Learning" by Ian Goodfellow et al., "Neural Networks and Deep Learning" by Michael Nielsen.
  - Online Courses: "Deep Learning" (Coursera), "PyTorch Fundamentals" (PyTorch Tutorials).

---

### **Additional Resources**

1. **Textbooks:**
   - "Python for Data Analysis" by Wes McKinney.
   - "The Elements of Statistical Learning" by Trevor Hastie et al.

2. **Online Platforms:**
   - **Kaggle:** Practice ML projects and competitions.
   - **GitHub:** Explore ML repositories and contribute to open-source projects.
   - **Towards Data Science:** Read articles and tutorials on ML and data science.

3. **Self-Study Topics:**
   - Mathematics refresher courses (Coursera, edX).
   - Python programming challenges (LeetCode, HackerRank).
   - ML algorithm implementations (GitHub, Kaggle).

---

### **Timeline**

| **Month** | **Activity**                             |
|-----------|-----------------------------------------|
| 1-2       | Complete Linear Algebra and Calculus courses. |
| 3-4       | Strengthen Python skills with CSci 343 and CSci 356. |
| 5-6       | Take CSci 443 (Advanced Data Science) and CSci 543 (Data Mining). |
| 7-9       | Enroll in CSci 632 (Machine Learning) and work on ML projects. |
| 10-12     | (Optional) Take Engr 691 (Deep Learning) and apply advanced techniques to biological data. |

---

### **Conclusion**

This learning plan is designed to systematically address the student’s knowledge gaps and prepare them for success in the Machine Learning course. By building a strong foundation in mathematics, programming, and ML theory, the student
will be well-equipped to apply ML techniques to biological research and advance their career in this interdisciplinary field.
(local LLM) azakaria@SOE-GGE-15067:~/Documents/DeepLearningProjects/Recommender_v3/learning-plan-recommender$ 
```

Figure 6: Final course recommendation and timeline for the student.

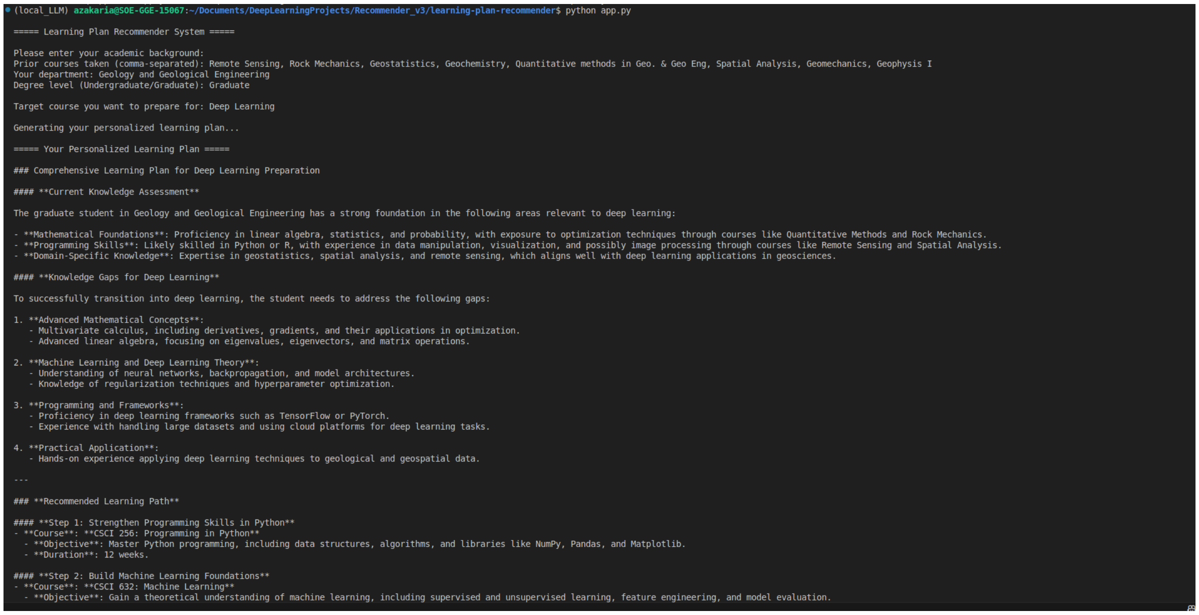


```
### **Recommended Learning Path**

#### **Step 1: Build Mathematical Foundations (3–4 Months)**
- **Courses:**
  - **Linear Algebra:** Enroll in an online course (e.g., "Linear Algebra for Machine Learning" on Coursera or edX).
  - **Calculus:** Take a course like "Calculus for Machine Learning" (Coursera or edX).
  - **Probability and Statistics:** Complete "Introduction to Probability and Statistics" (edX) or "Statistics for Data Science" (Coursera).
- **Resources:**
  - Textbooks: "Linear Algebra and Its Applications" by Gilbert Strang, "Calculus: Early Transcendentals" by James Stewart.
  - Online Tutorials: Khan Academy (Linear Algebra, Calculus), 3Blue1Brown (Essence of Linear Algebra, Essence of Calculus).

#### **Step 2: Strengthen Programming Skills (2–3 Months)**
- **Courses:**
  - **CSci 343: Fundamentals of Data Science** (covers data handling, visualization, and basic statistical analysis in Python).
  - **CSci 356: Data Structures in Python** (introduces advanced Python concepts and programming efficiency).
- **Self-Study:**
  - Practice with Pandas and NumPy through tutorials on DataCamp or Kaggle.
  - Explore scikit-learn and TensorFlow/PyTorch through official documentation and tutorials.
- **Projects:**
  - Implement basic ML algorithms (e.g., linear regression, k-nearest neighbors) from scratch using Python.

#### **Step 3: Introduction to Machine Learning (3–4 Months)**
- **Courses:**
  - **CSci 443: Advanced Data Science** (covers advanced data analysis and introduces ML algorithms).
  - **CSci 543: Data Mining** (focuses on clustering, classification, and real-world applications).
- **Resources:**
  - Textbooks: "An Introduction to Statistical Learning" by Gareth James et al., "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron.
  - Online Courses: "Introduction to Machine Learning" (Coursera), "Data Mining Specialization" (Coursera).

#### **Step 4: Core Machine Learning Course (4–5 Months)**
- **Course:**
  - **CSci 632: Machine Learning** (covers core ML algorithms, deep learning, and evaluation metrics).
- **Projects:**
  - Work on end-to-end ML projects using biological datasets (e.g., classify species, predict protein structures).
  - Participate in Kaggle competitions or contribute to open-source ML projects.
- **Resources:**
  - Textbooks: "Pattern Recognition and Machine Learning" by Christopher Bishop, "Deep Learning" by Ian Goodfellow et al.
  - Online Courses: "Machine Learning" (Coursera) by Andrew Ng, "Deep Learning Specialization" (Coursera).

#### **Step 5: Advanced Topics in Machine Learning (Optional – 3–4 Months)**
- **Course:**
  - **Engr 691: Special Topics in Engineering Science (Deep Learning - Graduate)** (covers advanced deep learning concepts and applications).
- **Projects:**
  - Apply deep learning to biological data (e.g., image recognition in neuroimaging, sequence analysis in genomics).
- **Resources:**
  - Textbooks: "Deep Learning" by Ian Goodfellow et al., "Neural Networks and Deep Learning" by Michael Nielsen.
  - Online Courses: "Deep Learning" (Coursera), "PyTorch Fundamentals" (PyTorch Tutorials).

---

### **Additional Resources**

1. **Textbooks:**
   - "Python for Data Analysis" by Wes McKinney.
   - "The Elements of Statistical Learning" by Trevor Hastie et al.
```

Figure 5: Knowledge Assessment of the student.

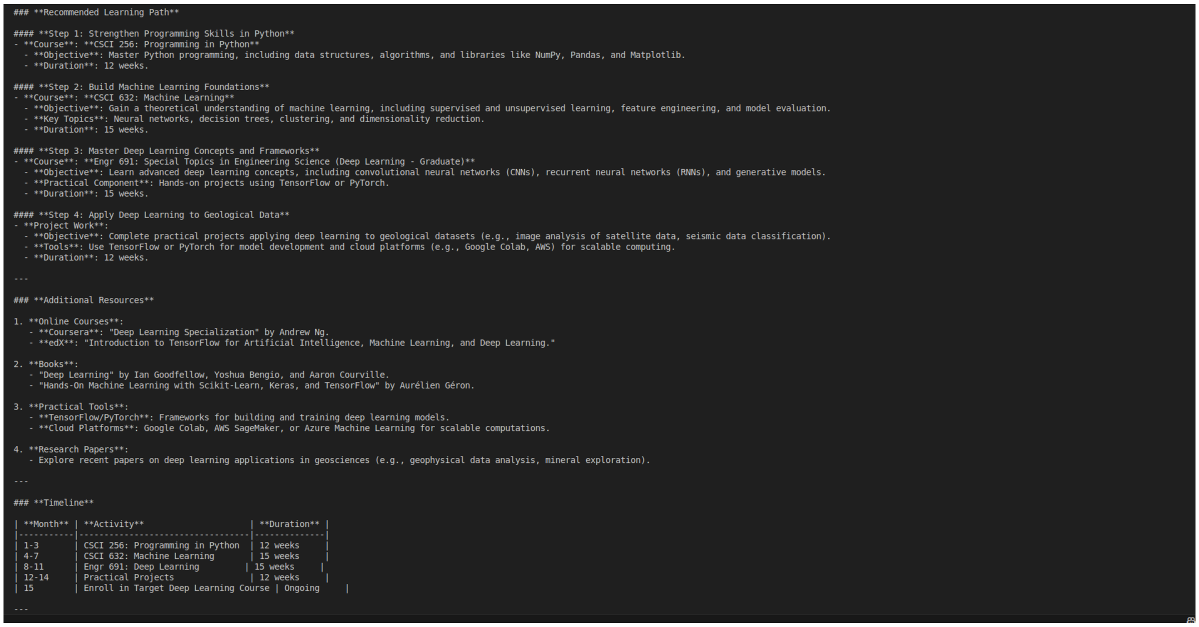
For this student, the recommendation system justly includes taking mathematical course in algebra and calculus as well as programming courses before enrolling in Machine learning. In addition, it also recommends taking deep learning as an option after taking the targeted machine learning class.

- Profile 3: Graduate student in Geological Engineering targeting to take the deep learning course.



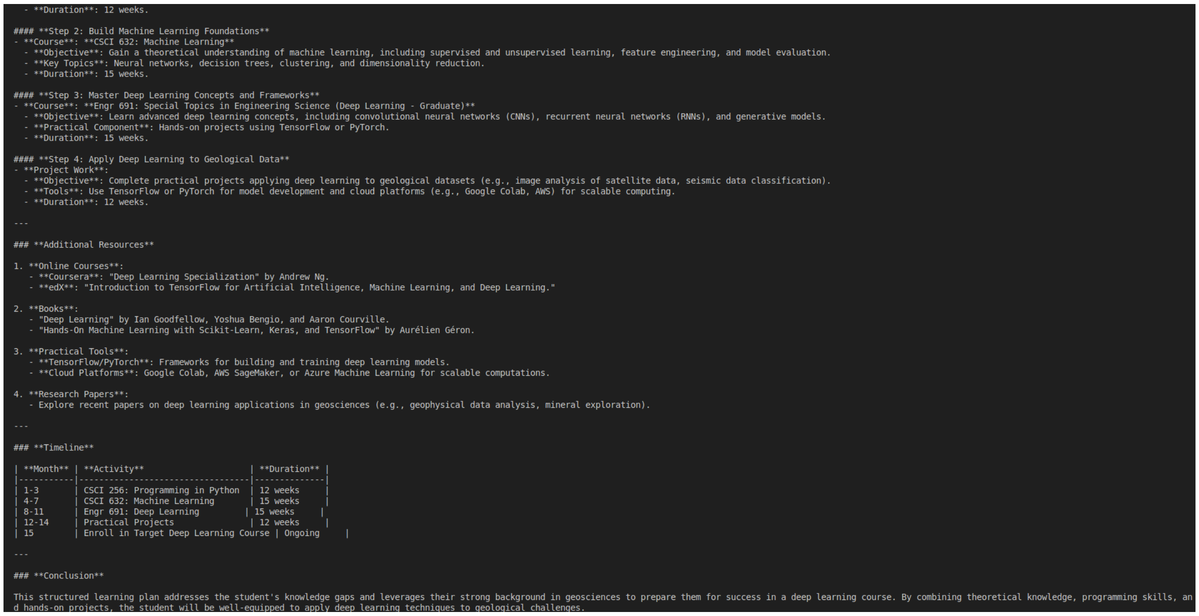
```
● (local_LLM) azakaria@SOE-GGE-15067:~/Documents/DeepLearningProjects/Recommender_v3/learning-plan-recommender$ python app.py

===== Learning Plan Recommender System =====

Please enter your academic background:
Prior courses taken (comma-separated): Remote Sensing, Rock Mechanics, Geostatistics, Geochemistry, Quantitative methods in Geo. & Geo Eng, Spatial Analysis, Geomechanics, Geophysis I
Your department: Geology and Geological Engineering
Degree level (Undergraduate/Graduate): Graduate

Target course you want to prepare for: Deep Learning

Generating your personalized learning plan...

===== Your Personalized Learning Plan =====

### Comprehensive Learning Plan for Deep Learning Preparation

#### **Current Knowledge Assessment**

The graduate student in Geology and Geological Engineering has a strong foundation in the following areas relevant to deep learning:

- **Mathematical Foundations**: Proficiency in linear algebra, statistics, and probability, with exposure to optimization techniques through courses like Quantitative Methods and Rock Mechanics.
- **Programming Skills**: Likely skilled in Python or R, with experience in data manipulation, visualization, and possibly image processing through courses like Remote Sensing and Spatial Analysis.
- **Domain-Specific Knowledge**: Expertise in geostatistics, spatial analysis, and remote sensing, which aligns well with deep learning applications in geosciences.

#### **Knowledge Gaps for Deep Learning**

To successfully transition into deep learning, the student needs to address the following gaps:

1. **Advanced Mathematical Concepts**:
   - Multivariate calculus, including derivatives, gradients, and their applications in optimization.
   - Advanced linear algebra, focusing on eigenvalues, eigenvectors, and matrix operations.

2. **Machine Learning and Deep Learning Theory**:
   - Understanding of neural networks, backpropagation, and model architectures.
   - Knowledge of regularization techniques and hyperparameter optimization.

3. **Programming and Frameworks**:
   - Proficiency in deep learning frameworks such as TensorFlow or PyTorch.
   - Experience with handling large datasets and using cloud platforms for deep learning tasks.

4. **Practical Application**:
   - Hands-on experience applying deep learning techniques to geological and geospatial data.

---

### **Recommended Learning Path**

#### **Step 1: Strengthen Programming Skills in Python**
- **Course**: **CSCI 256: Programming in Python**
  - **Objective**: Master Python programming, including data structures, algorithms, and libraries like NumPy, Pandas, and Matplotlib.
  - **Duration**: 12 weeks.

#### **Step 2: Build Machine Learning Foundations**
- **Course**: **CSCI 632: Machine Learning**
  - **Objective**: Gain a theoretical understanding of machine learning, including supervised and unsupervised learning, feature engineering, and model evaluation.
```

Figure 7: Essential information of the student.



```
### **Recommended Learning Path**

#### **Step 1: Strengthen Programming Skills in Python**
- **Course**: **CSCI 256: Programming in Python**
  - **Objective**: Master Python programming, including data structures, algorithms, and libraries like NumPy, Pandas, and Matplotlib.
  - **Duration**: 12 weeks.

#### **Step 2: Build Machine Learning Foundations**
- **Course**: **CSCI 632: Machine Learning**
  - **Objective**: Gain a theoretical understanding of machine learning, including supervised and unsupervised learning, feature engineering, and model evaluation.
  - **Key Topics**: Neural networks, decision trees, clustering, and dimensionality reduction.
  - **Duration**: 15 weeks.

#### **Step 3: Master Deep Learning Concepts and Frameworks**
- **Course**: **Engr 691: Special Topics in Engineering Science (Deep Learning - Graduate)**
  - **Objective**: Learn advanced deep learning concepts, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative models.
  - **Practical Component**: Hands-on projects using TensorFlow or PyTorch.
  - **Duration**: 15 weeks.

#### **Step 4: Apply Deep Learning to Geological Data**
- **Project Work**:
  - **Objective**: Complete practical projects applying deep learning to geological datasets (e.g., image analysis of satellite data, seismic data classification).
  - **Tools**: Use TensorFlow or PyTorch for model development and cloud platforms (e.g., Google Colab, AWS) for scalable computing.
  - **Duration**: 12 weeks.

---

### **Additional Resources**

1. **Online Courses**:
   - **Coursera**: "Deep Learning Specialization" by Andrew Ng.
   - **edX**: "Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning."

2. **Books**:
   - "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville.
   - "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron.

3. **Practical Tools**:
   - **TensorFlow/PyTorch**: Frameworks for building and training deep learning models.
   - **Cloud Platforms**: Google Colab, AWS SageMaker, or Azure Machine Learning for scalable computations.

4. **Research Papers**:
   - Explore recent papers on deep learning applications in geosciences (e.g., geophysical data analysis, mineral exploration).

---

### **Timeline**

| **Month** | **Activity**                    | **Duration** |
|-----------|---------------------------------|--------------|
| 1-3       | CSCI 256: Programming in Python  | 12 weeks     |
| 4-7       | CSCI 632: Machine Learning       | 15 weeks     |
| 8-11      | Engr 691: Deep Learning         | 15 weeks     |
| 12-14     | Practical Projects               | 12 weeks     |
| 15        | Enroll in Target Deep Learning Course | Ongoing     |

---
```

Figure 8: Knowledge Assessment of the student.



```
  - **Duration**: 12 weeks.

#### **Step 2: Build Machine Learning Foundations**
- **Course**: **CSCI 632: Machine Learning**
  - **Objective**: Gain a theoretical understanding of machine learning, including supervised and unsupervised learning, feature engineering, and model evaluation.
  - **Key Topics**: Neural networks, decision trees, clustering, and dimensionality reduction.
  - **Duration**: 15 weeks.

#### **Step 3: Master Deep Learning Concepts and Frameworks**
- **Course**: **Engr 691: Special Topics in Engineering Science (Deep Learning - Graduate)**
  - **Objective**: Learn advanced deep learning concepts, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and generative models.
  - **Practical Component**: Hands-on projects using TensorFlow or PyTorch.
  - **Duration**: 15 weeks.

#### **Step 4: Apply Deep Learning to Geological Data**
- **Project Work**:
  - **Objective**: Complete practical projects applying deep learning to geological datasets (e.g., image analysis of satellite data, seismic data classification).
  - **Tools**: Use TensorFlow or PyTorch for model development and cloud platforms (e.g., Google Colab, AWS) for scalable computing.
  - **Duration**: 12 weeks.

---

### **Additional Resources**

1. **Online Courses**:
   - **Coursera**: "Deep Learning Specialization" by Andrew Ng.
   - **edX**: "Introduction to TensorFlow for Artificial Intelligence, Machine Learning, and Deep Learning."

2. **Books**:
   - "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville.
   - "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron.

3. **Practical Tools**:
   - **TensorFlow/PyTorch**: Frameworks for building and training deep learning models.
   - **Cloud Platforms**: Google Colab, AWS SageMaker, or Azure Machine Learning for scalable computations.

4. **Research Papers**:
   - Explore recent papers on deep learning applications in geosciences (e.g., geophysical data analysis, mineral exploration).

---

### **Timeline**

| **Month** | **Activity**                     | **Duration** |
|-----------|----------------------------------|--------------|
| 1-3       | CSCI 256: Programming in Python  | 12 weeks     |
| 4-7       | CSCI 632: Machine Learning       | 15 weeks     |
| 8-11      | Engr 691: Deep Learning         | 15 weeks     |
| 12-14     | Practical Projects               | 12 weeks     |
| 15        | Enroll in Target Deep Learning Course | Ongoing     |

---

### **Conclusion**

This structured learning plan addresses the student's knowledge gaps and leverages their strong background in geosciences to prepare them for success in a deep learning course. By combining theoretical knowledge, programming skills, an
d hands-on projects, the student will be well-equipped to apply deep learning techniques to geological challenges.
```

Figure 9: Final course recommendation and timeline for the student.

> For this student, the recommendation system reasonably identifies the student's prior knowledge in linear algebra, statistics, and probability. It also recommends taking programming courses before enrolling in deep learning. Using the course catalog (RAG system), it also correctly identifies deep learning as an ongoing course this semester.

The code for this project can be found on GitHub as well as the GIF of these sample runs