МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Систем обработки информации и управления»

ОТЧЕТ

**Лабораторная работа №\_\_3\_\_**

по дисциплине«ИИ в задачах бизнес-аналитики»

Тема: «Методы адаптивного переключения и поддержки

множественного мировоззрения в системах

искусственного интеллекта»

ИСПОЛНИТЕЛЬ: \_\_Журавлев Н. В.\_\_

ФИО

группа ИУ5-44М \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

"17"\_\_апреля\_\_\_2025 г.

ПРЕПОДАВАТЕЛЬ: \_\_\_\_Сухобоков А.В.\_\_\_\_\_

ФИО

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

"17"\_апреля\_\_\_2025 г.

Москва - 2025

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Постановка задачи

1. Скачать и установить SWI Prolog.

2. Добавить через любой текстовый редактор несколько новых фактов и правил к каждому модулю (db1, db2, db3).

3. Сделать запросы на основе нескольких старых фактов, правил и новых фактов, правил с использованием разных конструкций findall(), ask().

4. Зарегистрироваться на https://mistral.ai/ или вручную установить и использовать ollama model.

5 Написать скрипт – программу (python или любую удобную для вас, учитывайте наличие библиотеки mistralai) способную:

1. Добавление новых фактов в формате Prolog с проверкой синтаксиса.

2. Удаление существующих фактов с подтверждением.

3. Поддержка трех типов баз: научные, мифологические, религиозные факты.

4. Автоматическая генерация Prolog-запросов из вопросов на естественном языке (с использованием Mistral AI).

5. Запуск запросов в SWI-Prolog и обработка результатов.

6. Преобразование формальных ответов Prolog в понятные формулировки на русском языке.

# Ход работы

С официального сайта необходимо скачать SWI Prolog.

После чего в файлах каждого модуля были добавлены правила и факты, часть которых представлена на рис. 1, рис. 2, рис. 3.

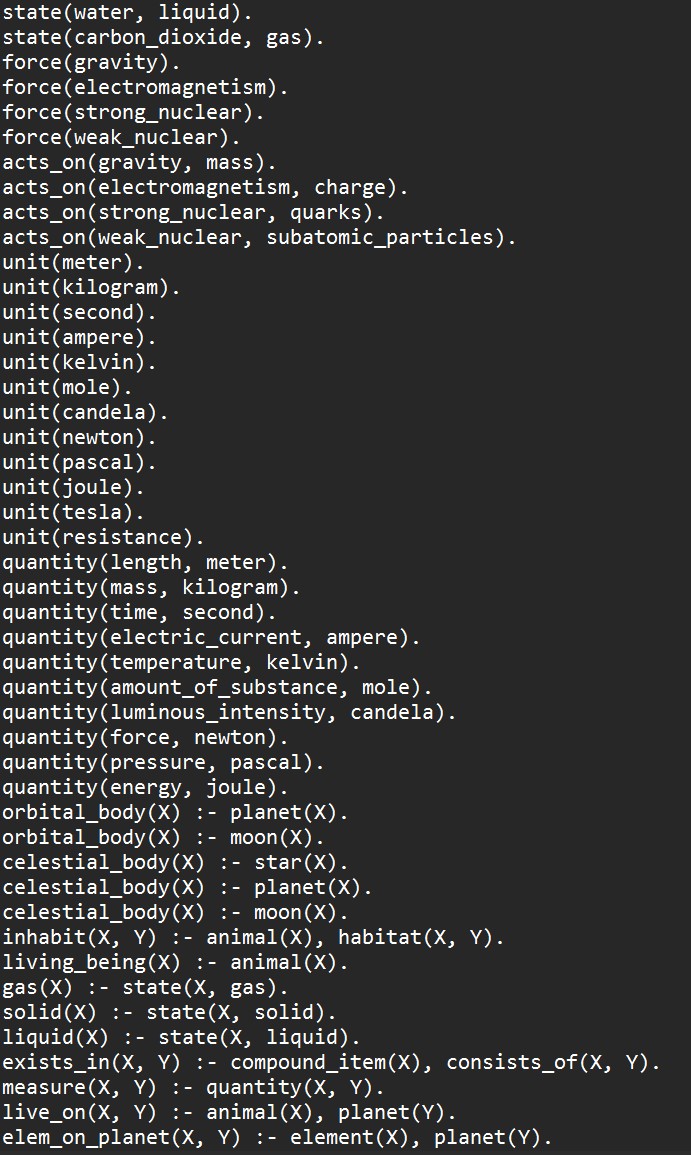


Рисунок 1. Фрагмент базы знаний научного мировоззрения

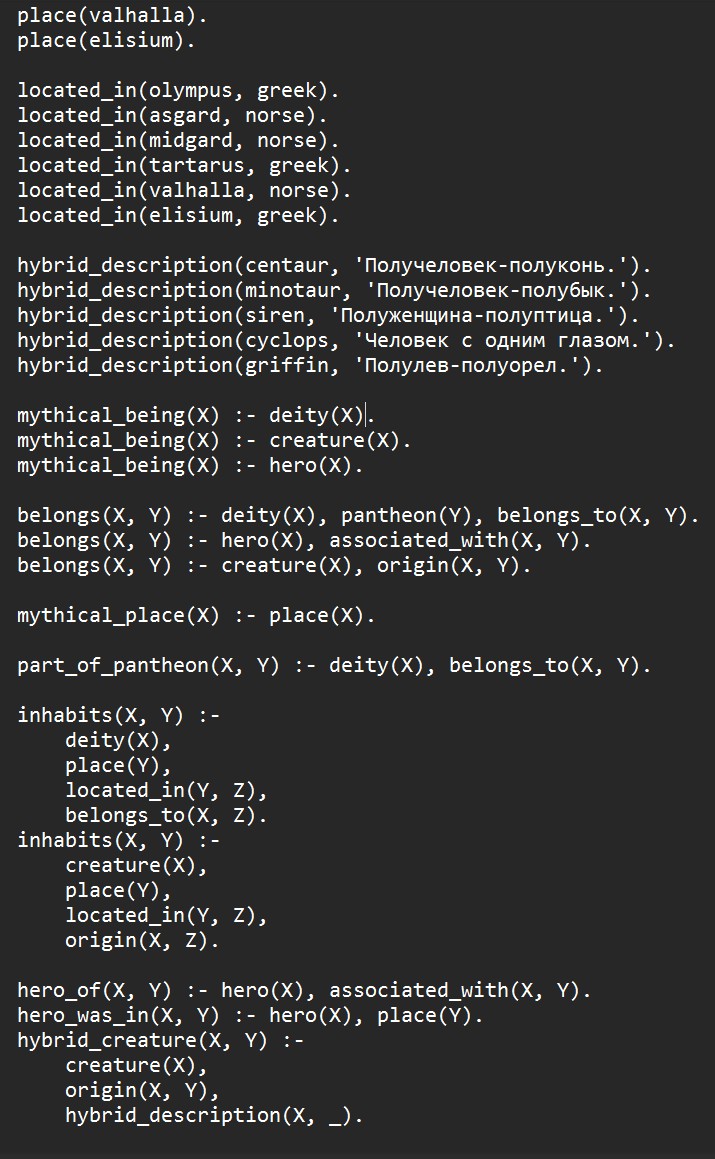


Рисунок 2. Фрагмент базы знаний мифического мировоззрения

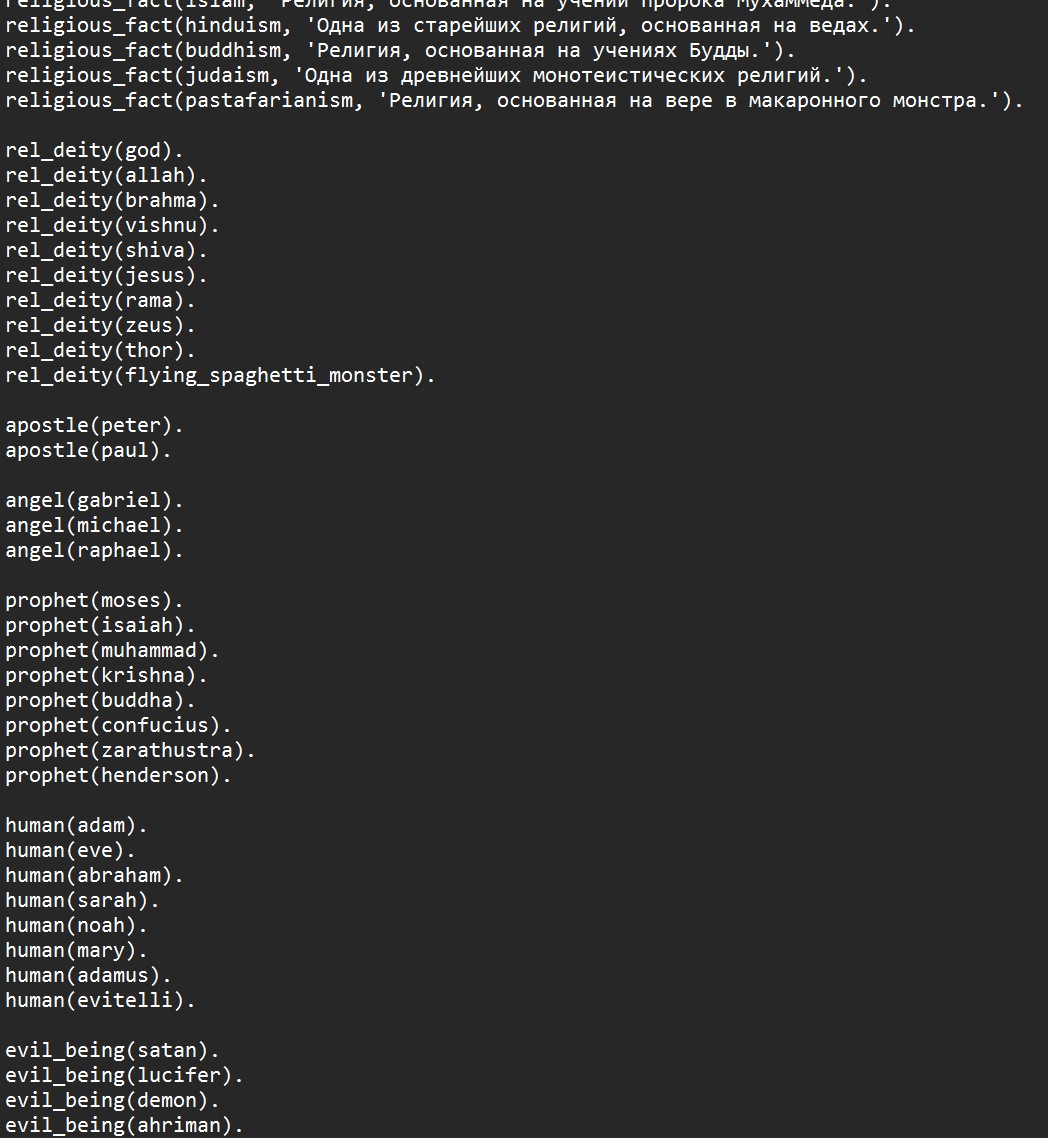


Рисунок 3. Фрагмент базы знаний религиозного мировоззрения

В начале необходимо установить Apache Superset. Для этого исполняются следующие команды:

Всего 9 запросов по 3 к каждому мировоззрению:

1) scientific\_ask(gravity).

Результат:

Сила притяжения между двумя телами.

2) scientific\_ask(valhalla).

Результат:

Молекула, хранящая генетическую информацию.

3) findall(Animal, (animal(Animal), breathe(Animal, air)), AirBreathingAnimals).

Результат:

AirBreathingAnimals = [human, dog, cat, elephant, dolphin].

4) findall(Element, elem\_on\_planet(Element, earth), EarthElements).

Результат:

EarthElements = [hydrogen, helium, lithium, beryllium, boron, carbon, nitrogen, oxygen, fluorine|...].

5) mythological\_ask(valhalla).

Результат:

Рай для павших воинов в скандинавской мифологии.

6) mythological\_ask(loki).

Результат:

Скандинавский бог хитрости и обмана.

7) findall(Child, parent(zeus, Child), ZeusChildren).

Результат:

ZeusChildren = [ares, athena, apollo].

8) findall(X, (hero(X), hero\_was\_in(X, valhalla)), Descs).

Результат:

Descs = [hercules, theseus, achilles, odysseus, sigurd, beowulf].

9) religious\_ask(christianity).

Результат:

Религия, основанная на учении Иисуса Христа.

10) religious\_ask(pastafarianism).

Результат:

Религия, основанная на вере в макаронного монстра.

11) findall(Scripture, rel\_contains(Scripture, new\_testament), ScriptureParts).

Результат:

ScriptureParts = [bible].

12) findall(Ritual, god\_wants(jesus, Ritual), Godwants).

Результат:

findall(Ritual, god\_wants(jesus, Ritual), Godwants).

Godwants = [prayer, fasting, pilgrimage, sacrifice, circumcision, langar, zen\_meditation, yasna].

Далее для взаимодействия с prolog на естественном языке была и редактирования баз знаний была написана программа на языке python, представленная на листинге 1.

Результат добавления новых фактов в формате Prolog на рис 4.

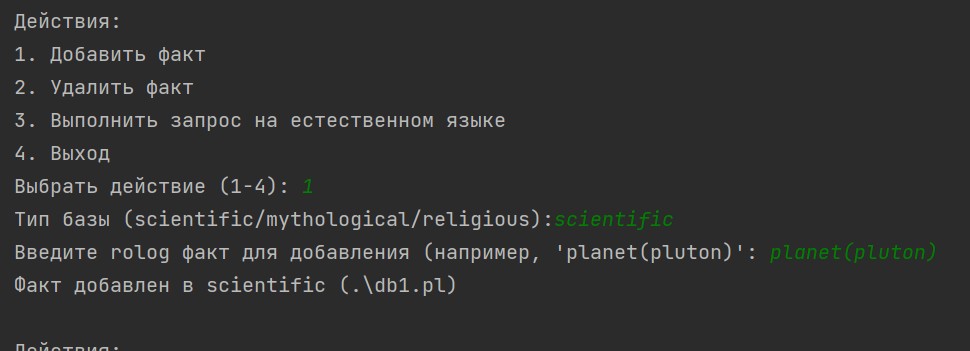


Рисунок 4. Результат добавления факта

Результат удаление существующих фактов представлен на рис 5.

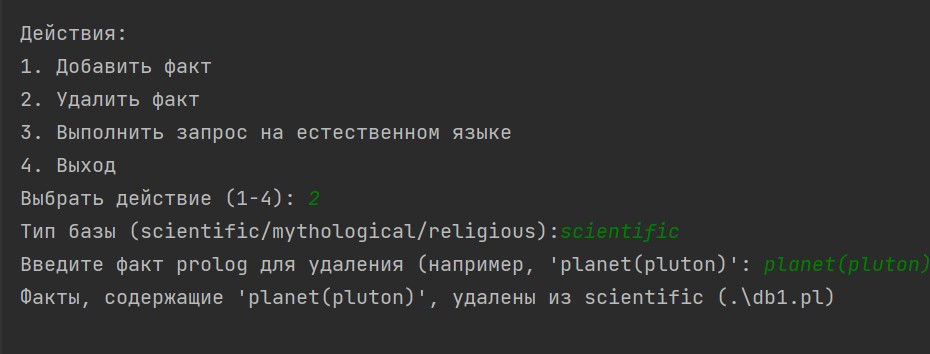


Рисунок 5. Результат удаления факта

Пример вывода запроса программы представлен на рис. 6.

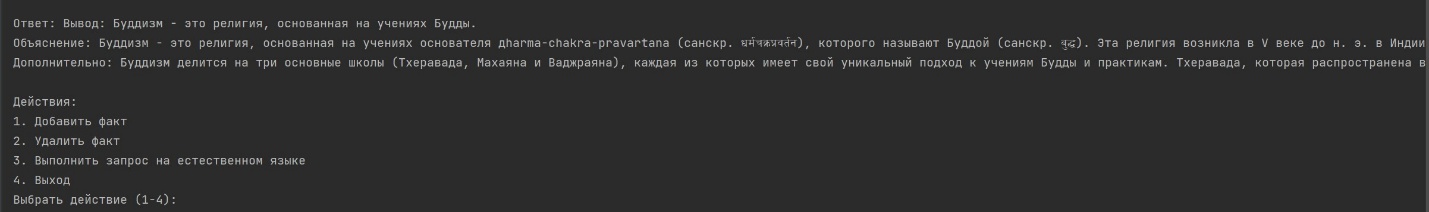


Рисунок *6*. Результат выполнения запроса на естественном языке

Листинг 1. Программа на языке python

|  |
| --- |
| import os  import re  from mistralai.client import MistralClient  from mistralai.models.chat\_completion import ChatMessage  import subprocess  class PrologKnowledgeBaseManager:  def \_\_init\_\_(self):  # Initialize Mistral AI client (replace with your API key)  mistral\_api\_key = "rVNC4BrYI33WOdvawnAV7mm2Fj1ADtMV"  self.mistral\_client = MistralClient(api\_key=mistral\_api\_key)  # Knowledge base files  self.knowledge\_bases = {  "scientific": "db1.pl",  "mythological": "db2.pl",  "religious": "db3.pl"  }  self.db1\_prompt = self.\_generate\_db\_prompt('scientific')  self.db2\_prompt = self.\_generate\_db\_prompt('mythological')  self.db3\_prompt = self.\_generate\_db\_prompt('religious')  def \_generate\_db\_prompt(self, db\_type):  """Генерация промта для конкретной базы данных"""  db\_path = os.path.join('.', self.knowledge\_bases[db\_type])  if not os.path.exists(db\_path):  return f"База данных {db\_type} не найдена по пути {db\_path}"  # Чтение структуры БД  with open(db\_path, 'r', encoding='utf-8') as f:  db\_content = f.read()  # Запрос к Mistral для генерации промта  prompt = f"""  Проанализируй структуру Prolog-базы данных и создай подробный промт для преобразования  естественно-языковых вопросов в Prolog-запросы. База относится к домену: {db\_type}.  Структура базы:  ```prolog  {db\_content}  ```  Требования к промту:  1. Должен содержать примеры предикатов из этой базы  2. Должен различать типы запросов (факты, правила, поиск)  3. Должен учитывать специфику домена ({db\_type})  4. Должен содержать шаблоны для разных типов вопросов  """  return prompt  def add\_fact(self, kb\_type: str, fact: str):  db\_path = os.path.join('.', self.knowledge\_bases[kb\_type])  try:  with open(db\_path, 'a') as f:  f.write(f"\n{fact}")  return f"Факт добавлен в {kb\_type} ({db\_path})"  except Exception as e:  return f"Ошибка: {str(e)}"  def delete\_fact(self, kb\_type: str, fact: str):  db\_path = os.path.join('.', self.knowledge\_bases[kb\_type])  try:  with open(db\_path, 'r') as f:  lines = f.readlines()  new\_lines = [l for l in lines if fact not in l]  with open(db\_path, 'w') as f:  f.writelines(new\_lines)  return f"Факты, содержащие '{fact}', удалены из {kb\_type} ({db\_path})"  except Exception as e:  return f"Ошибка: {str(e)}"  def get\_db\_prompt(self, db\_type):  """Получение сохраненного промта для базы"""  prompts = {  'scientific': self.db1\_prompt,  'mythological': self.db2\_prompt,  'religious': self.db3\_prompt  }  return prompts.get(db\_type, "Неизвестный тип базы данных")  def generate\_prolog\_query(self, natural\_language\_question: str, db\_type) -> str:  """Use Mistral AI to convert natural language to Prolog query"""  base\_prompt = self.get\_db\_prompt(db\_type)  prompt = f"""  {base\_prompt}  Преобразуй следующий вопрос в корректный Prolog-запрос для базы {db\_type}:  Вопрос: {natural\_language\_question}  Учитывай:  1. Если вопрос требует подтверждения (is/является), используй формат: `(предикат -> writeln('Да'); writeln('Нет'))`  2. Для описательных вопросов используй findall  3. Всегда указывай модуль (db1, db2 или db3) перед предикатами  Верни только Prolog-код без пояснений.  """  response = self.mistral\_client.chat(  model="mistral-tiny",  messages=[ChatMessage(role="user", content=prompt)]  )  return response.choices[0].message.content.strip()  def execute\_prolog\_query(self, query: str, db\_type) -> str:  """Execute a Prolog query using SWI-Prolog and return the raw results"""  db\_path = os.path.join('.', self.knowledge\_bases[db\_type])  main\_pl = os.path.join('.', 'main.pl')  try:  cmd = [  "swipl",  '-q',  '-s', main\_pl,  '-s', db\_path,  '-g', query,  '-t', 'halt'  ]  result = subprocess.run(  cmd,  capture\_output=True,  text=True,  timeout=10  )  if result.returncode != 0:  return f"Ошибка Prolog: {result.stderr}"  cleaned = re.sub(r'^%.\*\n?', '', result.stdout, flags=re.MULTILINE).strip()  if not cleaned:  return "Нет результатов"  if re.match(r'^(true|yes)\b', cleaned, re.I):  return "Да"  elif re.match(r'^(false|no)\b', cleaned, re.I):  return "Нет"  # Извлекаем множественные результаты  if '=' in cleaned:  return "\n".join(  f"{var.strip()} = {val.strip()}"  for var, val in re.findall(r'(\w+)\s\*=\s\*([^\n]+)', cleaned)  )  return cleaned  except subprocess.TimeoutExpired:  return "Запрос превысил время выполнения"  except Exception as e:  return f"Ошибка выполнения: {str(e)}"  def translate\_prolog\_response(self, prolog\_response: str, original\_question: str) -> str:  """Use Mistral AI to translate formal Prolog response to natural language"""  prompt = f"""  Ты - эксперт по объяснению результатов Prolog-запросов.  Вопрос: {original\_question}  Ответ системы: {prolog\_response}  Сгенерируй:  1. Краткий вывод (1 предложение)  2. Подробное объяснение (2-3 предложения)  3. Дополнительные факты (если уместно)  Формат:  Вывод: ...  Объяснение: ...  Дополнительно: ...  Весь ответ должен быть ТОЛЬКО на русском языке. Переведи все слова.  """  response = self.mistral\_client.chat(  model="mistral-tiny",  messages=[ChatMessage(role="user", content=prompt)]  )  return response.choices[0].message.content.strip()  def interactive\_mode(self):  while True:  print("\nДействия:")  print("1. Добавить факт")  print("2. Удалить факт")  print("3. Выполнить запрос на естественном языке")  print("4. Выход")  choice = input("Выбрать действие (1-4): ")  if choice == "1":  kb\_type = input("Тип базы (scientific/mythological/religious):")  if kb\_type not in manager.knowledge\_bases:  print("Неверный тип базы знаний")  continue  fact = input("Введите rolog факт для добавления (например, 'planet(pluton)': ")  message = self.add\_fact(kb\_type, fact)  print(message)  elif choice == "2":  kb\_type = input("Тип базы (scientific/mythological/religious):")  if kb\_type not in manager.knowledge\_bases:  print("Неверный тип базы знаний")  continue  fact = input("Введите факт prolog для удаления (например, 'planet(pluton)': ")  message = self.delete\_fact(kb\_type, fact)  print(message)  elif choice == "3":  kb\_type = input("Тип базы (scientific/mythological/religious):")  if kb\_type not in manager.knowledge\_bases:  print("Неверный тип базы знаний")  continue  question = input("Введите вопрос на естественном языке: ")  try:  # Generate Prolog query  prolog\_query = self.generate\_prolog\_query(question, kb\_type)  print(f"Сгенерированный Prolog-запрос: {prolog\_query}")  # Execute query  prolog\_response = self.execute\_prolog\_query(prolog\_query, kb\_type)  print(f"Ответ prolog: {prolog\_response}")  # Translate to natural language  natural\_response = self.translate\_prolog\_response(prolog\_response, question)  print(f"\nОтвет: {natural\_response}")  except Exception as e:  print(f"Ошибка выполнения запроса: {str(e)}")  elif choice == "4":  print("Exiting...")  break  else:  print("Invalid option, please try again.")  if \_\_name\_\_ == "\_\_main\_\_":  manager = PrologKnowledgeBaseManager()  manager.interactive\_mode() |

# Вывод

В ходе выполнения лабораторной работы были получены навыки взаимодействия с SWI Prolog и программное подключение к модели Mistral AI) и взаимодействие с ней.