**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное автономное образовательное учреждение высшего образования**

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**РУТ(МИИТ)**

**Институт управления и цифровых технологий**

Кафедра «Вычислительные системы, сети и информационная безопасность»

**Отчет По ПРОЕКТНОй ДЕЯТЕЛЬНОСТИ**

**НА тему:**

Направление: 09.03.01 Информатика и вычислительная техника

Профиль:Вычислительные системы и сети

Выполнили:  
студенты группы УВВ-111

Жданов А. И.

Подрез А. П.

Рассадин К. Ю.

Проверил:

Ассистент Желенкова М. Б.

**МОСКВА 2025**

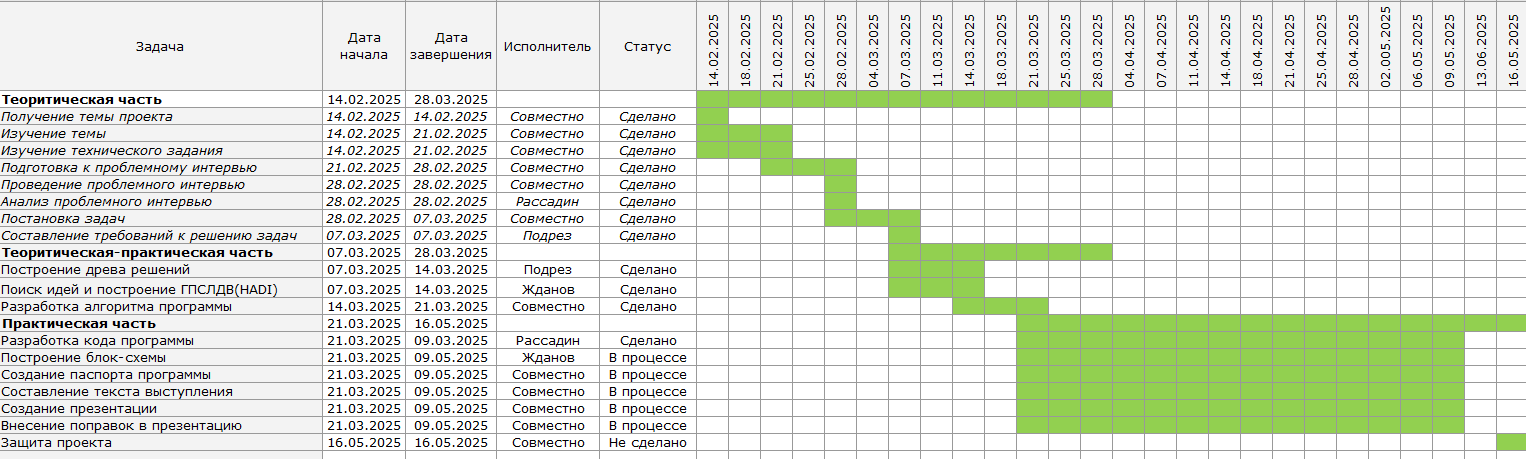


Таблица 1

| **Раздел** | **Рекомендации** |
| --- | --- |
| Название проекта | Определение опасных участков железных дорог |
| Проектный наставник | Желенкова Маргарита Борисовна. |
| Команда студенческого проекта | Рассадин Кирилл Юрьевич – Исполнитель практической части  Подрез Алексей Павлович – руководитель команды  Жданов Артём Иванович – Исполнитель теоретической части |
| Заказчик | ОАО “РЖД”, Начальник управления ЦБА |
| Проблема | Низкая эффективность идентификации потенциально опасных зон |
| Анализ аналогов | **Создать программу для определения границ опасных участков пути**  Купить мощное оборудование для проведения анализа  Написать программу на старом языке программирования |
| Описание реализованного решения | 1. Лёгкая в реализации и эффективная в работе 2. Не требует больших человеческих ресурсов 3. Дешёвая в обслуживании 4. Точные компьютерные рассчёты |
| Пользователи | ОАО “РЖД” |
| Материалы проекта | Программа представляет собой отрезки пути разбитые на определённые групы |
| Ссылка на репозиторий рабочих материалов проекта | [https://github.com/sntuew/project](https://github.com/llirik28/neuro-pr1) |
| Сроки исполнения (фактические) | 16 февраля 2025 – 12 мая 2025 |
| Затраченные ресурсы | Человеческие |
| Эксперты | Рудыка Наталья Алексеевна, Желенкова Маргарита Борисовна |

Блок-схема 1

import numpy as np  
  
  
  
def compute\_correlation\_matrix(file\_path):  
 data = np.genfromtxt(file\_path, delimiter=';')  
 X = data[:, 1:] if data.shape[1] > 1 else data  
 means = np.mean(X, axis=0)  
 deviations = X - means  
 cov\_matrix = np.dot(deviations.T, deviations) / (X.shape[0] - 1)  
 std\_devs = np.sqrt(np.diag(cov\_matrix))  
 corr\_matrix = cov\_matrix / np.outer(std\_devs, std\_devs)  
 return corr\_matrix  
  
  
def build\_correlation\_graph(corr\_matrix, threshold):  
 n = corr\_matrix.shape[0]  
 graph = {i: [] for i in range(n)}  
  
 for i in range(n):  
 for j in range(i + 1, n):  
 if abs(corr\_matrix[i, j]) >= threshold:  
 graph[i].append((j, corr\_matrix[i, j]))  
 graph[j].append((i, corr\_matrix[i, j]))  
  
 return graph  
  
  
def find\_connected\_components(graph):  
 visited = set()  
 components = []  
  
 for node in graph:  
 if node not in visited:  
 stack = [node]  
 component = []  
 while stack:  
 current = stack.pop()  
 if current not in visited:  
 visited.add(current)  
 component.append(current)  
 for neighbor, \_ in graph[current]:  
 if neighbor not in visited:  
 stack.append(neighbor)  
 components.append(component)  
  
 return components  
  
  
def group\_features(corr\_matrix, threshold):  
 graph = build\_correlation\_graph(corr\_matrix, threshold)  
 components = find\_connected\_components(graph)  
 return components  
  
file\_path = "testA.csv"  
try:  
 np.set\_printoptions(  
 precision=3,  
 suppress=True,  
 linewidth=100  
 )  
 corr\_matrix = compute\_correlation\_matrix(file\_path)  
 print("Корреляционная матрица:")  
 print(corr\_matrix)  
  
 threshold = 0.63  
  
  
 feature\_groups = group\_features(corr\_matrix, threshold)  
 print("\nГруппы признаков:")  
 for i, group in enumerate(feature\_groups, 1):  
 print(f"Группа {i}: {[x + 1 for x in group]}")  
  
except Exception as e:  
 print(f"Произошла ошибка: {e}")