Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет: Институт издательского дела и журналистики  
Кафедра «Информатика и информационные технологии»

Направление подготовки/ специальность: Информационные системы и технологии

ОТЧЕТ

по проектной практике

Студент: Федукина Мария Павловна Группа: 241-334

Место прохождения практики: Московский Политех, кафедра «информатика и информационные технологии»

Отчет принят с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Дата\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики: Рябчикова Анна Валерьевна

Москва 2025

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

1. Общая информация о проекте:

* Название проекта
* Цели и задачи проекта

1. Общая характеристика деятельности организации *(заказчика проекта)*

* Наименование заказчика
* Организационная структура
* Описание деятельности

1. Описание задания по проектной практике
2. Описание достигнутых результатов по проектной практике

ЗАКЛЮЧЕНИЕ *(выводы о проделанной работе и оценка ценности выполненных задач для заказчика)*

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

ПРИЛОЖЕНИЯ *(при необходимости)*

**ВВЕДЕНИЕ**

1. **Общая информация о проекте**

**Название проекта**

Название проекта «CharCreator» - производное от двух английских слов (character – персонаж, герой; creator – создатель). По задумке название должно быть простым в произношении и запоминании как русской, так и английской аудитории, а также отражать суть проекта.

**Цели и задачи проекта**

Главной целью проекта является создание функционирующего сайта со встроенным визуальным редактором внешности персонажа для DnD. Для выполнения данной цели необходимо не только четко выстроить цепочку задач, но и осознавать, что из себя представляет DnD. DnD – это ролевая настольная игра в жанре фэнтези, в которой есть игроки и мастер. Игроки под руководством мастера отправляются в выдуманное приключение, отыгрывая созданного ими персонажа. Для этого используются специальные листы персонажа, в которые записывается вся информация о нем, в том числе и его внешность. Таким образом, проект CharCreator это визуальный редактор внешности персонажа, изначально специализирующийся именно на игроках DnD. Для реализации данного проекта был поставлен ряд задач:

* Сбор и анализ информации:
* Опрос ЦА;
* Анализ трендов и конкурентов;
* Создание карты пути пользователя;
* Создание и распределение технического задания (далее, ТЗ):
* ТЗ для дизайнеров проекта;
* ТЗ для программистов;
* Работа над дизайном:
* Определение цветового решения;
* Создание логотипа;
* Создание макета сайта (как цифрового, так и бумажного);
* Создание дизайна персонажей;
* Работа над технической частью:
* Определение языка программирования, домена и т.п.;
* Разработка сайта и визуального редактора;
* Работа над социальными сетями:
* Создание группы ВКонтакте;
* Привлечение аудитории;
* Оформление постов, работа над их регулярным выходом и поиск нового контента.

**Актуальность проекта**

Актуальность проекта обусловлена ростом популярности настольной игры DnD. Рост популярности настольной игры приводит к притоку новых игроков, которые не привыкли к традиционной записи статистики своего игрового персонажа и поиску информации об игре в объемной Книге Игрока или Мастера, поэтому стремятся его оптимизировать: именно благодаря этому существуют сейчас такие сайты как «Long Story Short» и «dnd.su». Однако их функционал не подразумевает визуального редактора внешности персонажа. Существуют другие сайты, содержащие в себе подобный функционал, однако, по тем или иным причинам они недостаточно соответствуют требованиям игроков, что доказал проведенный анализ конкурентов и трендов. Именно поэтому создание визуального редактора внешности персонажа является актуальной проблемой.

1. **Общая характеристика деятельности организации (заказчика проекта)**

**Наименование заказчика**

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский Политехнический Университет».

**Организационная структура**

Организационная структура университета включает в себя несколько ключевых компонентов:

* административное управление (ректорат, отделы по учебной и научной деятельности);
* структурные подразделения – факультеты и институты, формирующие образовательный и исследовательский контент;
* центр дополнительного профессионального образования, обеспечивающий реализацию программ вне рамок основной учебной нагрузки.

**Описание деятельности**

Московский Политех осуществляет образовательную деятельность в рамках программ бакалавриата, магистратуры и дополнительного профессионального образования. Университет предлагает широкий спектр учебных направлений – от технических и инженерных до гуманитарных и цифровых, а также развивает курсы переподготовки и повышения квалификации. Одной из задач, стоящих перед университетом, является необходимость создания более эффективной, централизованной системы администрирования программ дополнительного образования, что открывает перспективы для цифровизации и автоматизации учебных процессов.

1. **Описание задания по проектной практике**

В рамках проектной практики было поставлено задание разработать веб-продукт, тематически связанный с основным проектом, который выполнялся на «Проектной деятельности». Практика включала как базовую, так и вариативную части и предусматривала не только техническую реализацию, но и оформление сопроводительной документации.

Проект выполнялся индивидуально. Вся работа – от планирования структуры до наполнения контентом и публикации – велась мной самостоятельно, с использованием репозитория на GitHub (<https://github.com/uneverdreamed/project-practice-site>), что обеспечивало контроль версий и прозрачность хода разработки.

Базовая часть задания включала:

* инициализацию и настройку репозитория проекта;
* создание сайта с помощью статического генератора Hugo;
* реализацию нескольких разделов, включая главную страницу, страницу «О проекте», «Журнал» и «Ресурсы»;
* работу с шаблонами темы оформления (Ananke), адаптацию структуры сайта под цели проекта;
* размещение на сайте материалов проекта: презентации, графических ресурсов, игровых правил и ссылок;
* оформление проектной документации и итогового отчёта.

В процессе работы мной была выполнена настройка сайта на Hugo, подобрана и подключена тема оформления, организована структура контента (в том числе через каталоги content, static и layouts), а также подготовлены страницы с использованием синтаксиса Markdown. В разделе «Журнал» были опубликованы записи, отслеживающие ход разработки, в «Ресурсах» — выложены вспомогательные материалы, необходимые для ознакомления с проектом. Кроме того, были созданы главная страница с общей информацией о проекте и страница с описанием участников. Все страницы сопровождаются фотографиями и визуально понятным оформлением, соответствующим общей стилистике проекта.

Проект создавался с нуля и стал логическим продолжением работы, выполненной в течение года в рамках основного проекта «CharCreator». В результате была подготовлена полноценная веб-страница, выполняющая функцию цифровой презентации и хранилища материалов проекта.

Вариативная часть включала разработку простой нейросетевой модели на языке Python, способной предсказывать следующий ход противника в игре «Камень-ножницы-бумага» на основе истории предыдущих раундов. Цель заключалась в демонстрации базовых принципов машинного обучения: подготовки данных, построения модели, обучения и оценки её качества.

Работа выполнялась индивидуально. Основные этапы включали:

* **Формирование обучающего датасета**: были собраны и размечены реальные и синтетические данные, представляющие собой последовательности ходов игрока и противника. Дополнительно реализована функция генерации правдоподобных последовательностей на основе вероятностных правил, имитирующих поведение игрока.
* **Предобработка данных и кодирование признаков:** каждый ход преобразовывался в вектор в формате one-hot. На основе истории длиной N (в финальной версии – 5 раундов) формировались входные признаки, учитывающие как действия самого игрока, так и его оппонента.
* **Разработка архитектуры нейросети:** была построена простая двухслойная модель с одним скрытым слоем. Входной слой принимал кодированные данные, скрытый слой активировался функцией sigmoid, выходной – softmax с тремя нейронами (для «Камня», «Ножниц» и «Бумаги»). Использовалась кросс-энтропийная функция потерь и L2-регуляризация.
* **Реализация обучения модели с нуля:** все операции реализованы вручную без использования готовых фреймворков, что позволило глубже понять принципы работы нейросетей.
* **Проведение обучения и анализ результатов:** модель обучалась на 80% данных, а точность проверялась на отложенной тестовой выборке (20%). Итоговая точность на тесте составила около 41%, что существенно превышает случайное угадывание (33%). Также были построены график снижения ошибки и матрица неточностей.
* **Формулировка выводов и ограничений:** дальнейшее повышение точности потребует перехода к более сложным архитектурам расширения обучающего набора. Работа продемонстрировала базовые возможности ИИ в предсказании шаблонного поведения в игре и может быть использована как основа для дальнейших экспериментов.

1. **Описание достигнутых результатов по проектной практике**

В рамках выполнения проектной практики мной был разработан сайт проекта с использованием генератора статических сайтов Hugo. Это позволило не только ускорить процесс создания сайта, но и получить опыт работы с современными веб-инструментами. Создана и оформлена структура репозитория на GitHub, в том числе добавлен ключевой файл README.md, поясняющий содержание проекта.

Разработан полноценный сайт, включающий пять основных разделов:

* **Главная страница** (main) – краткое описание проекта и его цели;
* **О проекте** (about) – подробный текст о замысле, актуальности, этапах и результатах;
* **Участники** (team) – информация об авторах проекта;
* **Журнал** (blog) – записи, отражающие ход выполнения работы;
* **Ресурсы** (resources) – вспомогательные материалы, включая правила настольной игры, презентацию, ссылки на фигму и соцсети.

Каждая страница оформлена с использованием Markdown и интегрирована в структуру сайта Hugo. Визуально сайт построен на базе темы Ananke, поддерживающей адаптивность и простоту восприятия. Хотя не удалось полностью реализовать задуманный внешний вид, использование Hugo соответствовало методическим рекомендациям, в которых прямо указывается, что генераторы сайтов (в особенности Hugo) являются предпочтительным вариантом при наличии базовых навыков HTML/CSS.

В рамках вариативной части практики, была разработана простая нейросетевая модель на языке Python, предсказывающая следующий ход противника в игре «Камень-ножницы-бумага» на основе истории предыдущих раундов:

* + кодировка входных данных в формате one-hot (ходы игрока и соперника за 5 последних раундов);
  + двухслойная архитектура: скрытый слой с 16 нейронами и сигмоидой, выходной слой с функцией softmax;

Дополнительно были реализованы модули, обеспечивающие полноценную работу и оценку модели:

* генерация обучающих данных с ограничением на повтор ходов;
* обучение нейросети с нуля методом обратного распространения ошибки (backpropagation);
* визуализация процесса обучения (график ошибки), а также оценка точности предсказания и построение confusion matrix;
* демо-режим, позволяющий сыграть против обученной модели (отключено по умолчанию).

**Приобретенные навыки:**

1. **Работа с Git и GitHub**
   * Создание и структурирование репозитория проекта
   * Ведение истории изменений и работа с коммитами
   * Размещение проекта в удалённом репозитории
2. **Работа с генераторами сайтов (Hugo)**
   * Установка и настройка Hugo
   * Использование темы оформления (Ananke), адаптация шаблонов
   * Работа с Markdown и параметрами конфигурации (config.toml)
   * Создание структуры сайта и страниц на основе шаблонов
3. **Основы веб-разработки**
   * Написание и редактирование HTML и CSS
   * Интеграция медиа-контента (изображений, логотипов, фоновых изображений)
   * Попытка настройки визуальной части сайта (цветовая схема, фон и т.п.)
4. **Python-разработка**
   * Работа с массивами и матрицами при кодировании входных данных (one-hot)
   * Реализация собственной нейросети с нуля, включая прямой и обратный проход
   * Создание инструментов для анализа данных и визуализации результатов обучения (matplotlib)
5. **Документирование проекта**
   * Подготовка и оформление текстов для сайта
   * Структурирование информации в формате Markdown
   * Составление итогового отчета по проектной практике

**ЗАКЛЮЧЕНИЕ**

Проектная практика стала для меня важным этапом в профессиональном и личном развитии. Она позволила не только применить полученные ранее знания на практике, но и попробовать себя в роли разработчика полноценного цифрового продукта – от проектирования структуры до технической реализации и финального оформления.

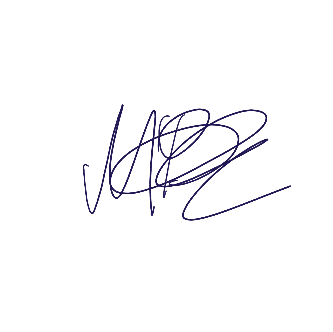
Работа над сайтом проекта дала возможность освоить генератор статических сайтов Hugo, научиться организовывать структуру проекта, работать с шаблонами и конфигурационными файлами, а также размещать контент с помощью языка разметки Markdown. Несмотря на то что мне не удалось в полной мере реализовать изначально задуманную визуальную концепцию сайта (в частности, настройку фона и цветовой схемы), я получила полезный опыт адаптации готовых тем и настройки внешнего вида с помощью CSS и параметров Hugo.

Особую ценность имела реализация вариативной части задания – нейросетевой модели на Python. Работа над ней потребовала глубокого погружения в основы машинного обучения, включая подготовку и обработку данных, разработку архитектуры модели и реализацию обучения с нуля. Это позволило не только лучше понять алгоритмы работы нейросетей, но и развить навыки работы с числовыми данными, логикой программирования и визуализацией результатов.

Также в рамках проекта я освоила базовые принципы работы с Git и GitHub, что обеспечило прозрачность и управляемость процесса разработки. Подготовка отчета и оформление сайта помогли систематизировать знания, улучшить навыки технического письма и визуального представления информации.

Несмотря на отсутствие командной работы, выполнение всех этапов проекта в одиночку дало ценный опыт самоуправления, планирования и решения нестандартных задач. Уверена, что этот проект станет важной вехой в моей учебной и будущей профессиональной деятельности. Полученные знания и навыки обязательно найдут применение в дальнейших образовательных и практических задачах, особенно в сфере разработки пользовательских интерфейсов и интеллектуальных систем.

Подтверждаю, что отчет выполнен лично и соответствует требованиям практики

**

Федукина Мария Павловна 23.05.2025 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*ФИО дата подпись*

**СПИСОК ЛИТЕРАТУРЫ**

1. Купер Нейт, Джи Ким. Как создать сайт: Пошаговое руководство для начинающих. — М.: Манн, Иванов и Фербер, 2016.
2. How Many People Play D&D in 2024? (User & Growth Stats) // Fiction Horizon. — URL: <https://fictionhorizon.com/how-many-people-play-dd/>
3. Hugo Quick Start Guide — Официальная документация Hugo. — URL: https://gohugo.io/getting-started/quick-start/
4. Hugo Documentation — Полная документация по Hugo. — URL: https://gohugo.io/documentation/
5. Ananke Theme for Hugo — GitHub репозиторий темы Ananke. — URL: <https://github.com/theNewDynamic/gohugo-theme-ananke>
6. Pro Git (Scott Chacon and Ben Straub). — 2nd Edition. Apress, 2014. — URL: <https://git-scm.com/book/en/v2>
7. Markdown Guide — Полное руководство по синтаксису Markdown. — URL: https://www.markdownguide.org/basic-syntax/
8. Matplotlib Documentation — Официальная документация библиотеки Matplotlib. — URL: https://matplotlib.org/stable/index.html

ПРИЛОЖЕНИЯ

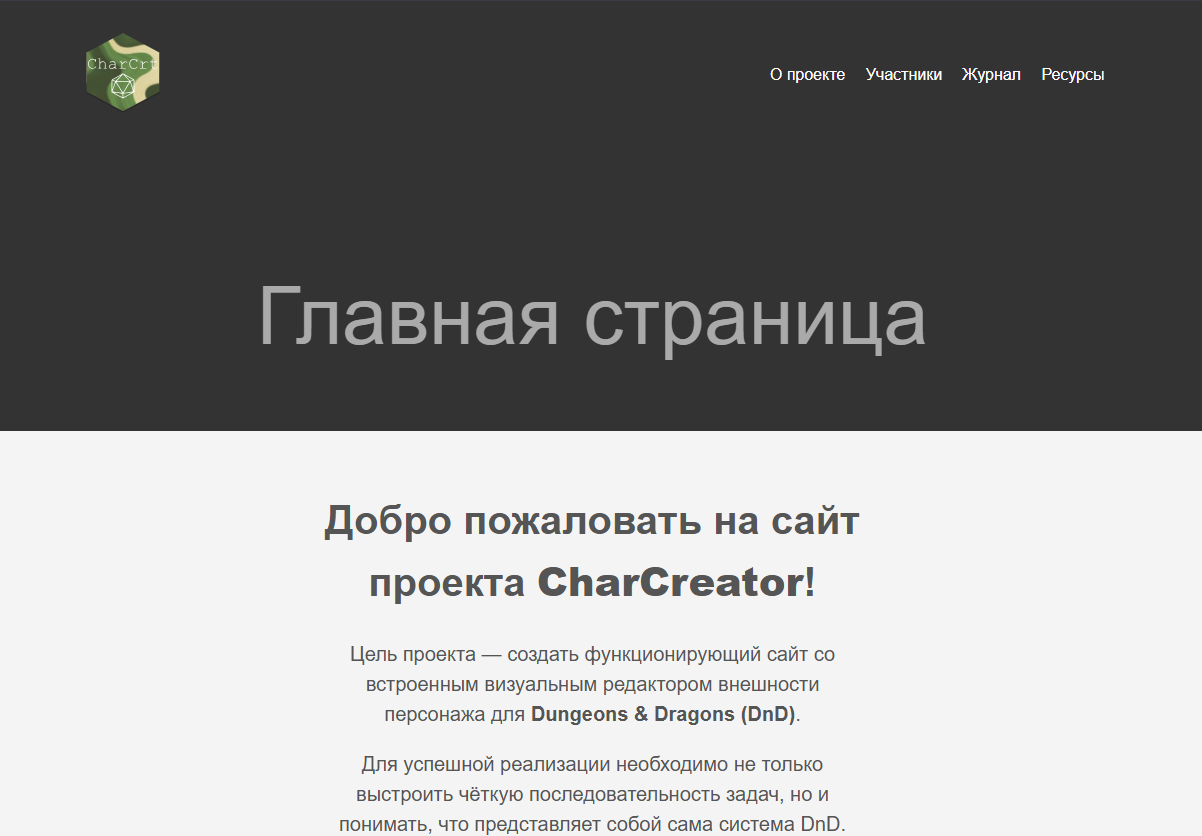


Рисунок 1 - страница main 1

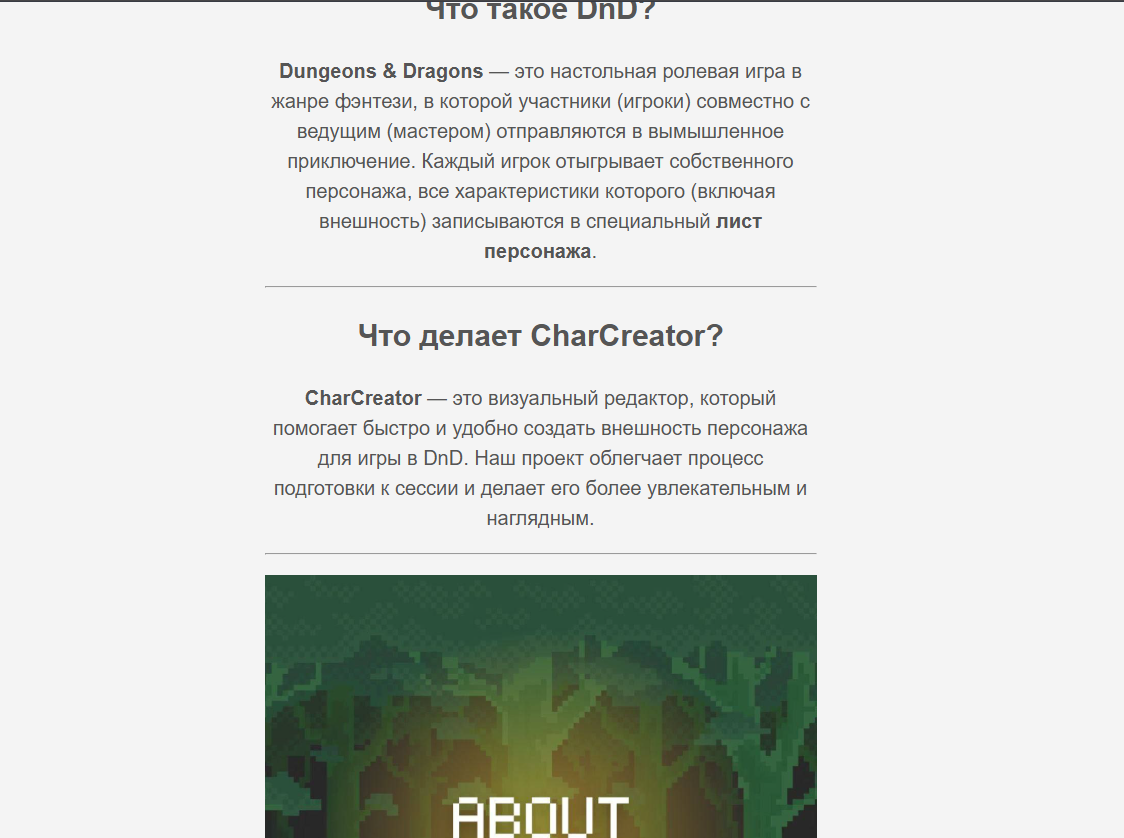


Рисунок 2 - страница main 2

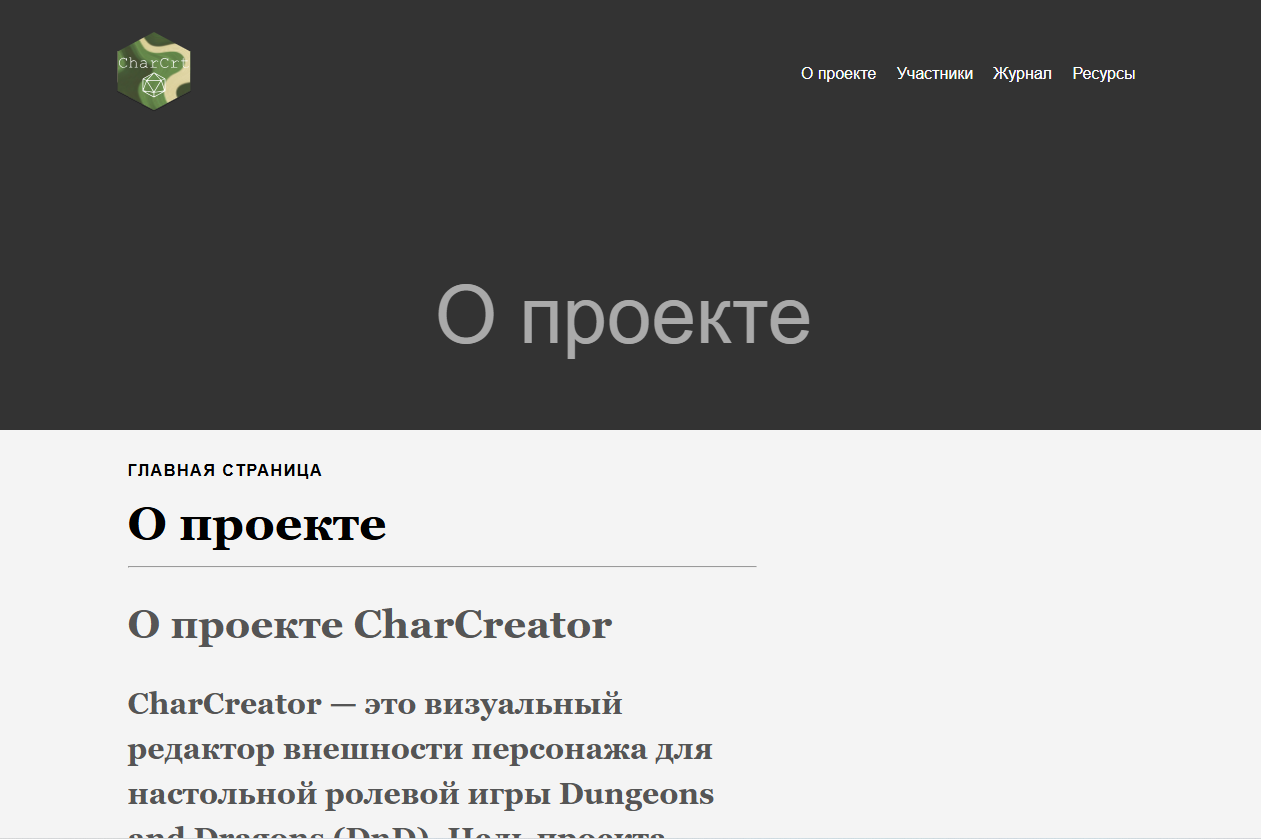


Рисунок 3 - страница about 1

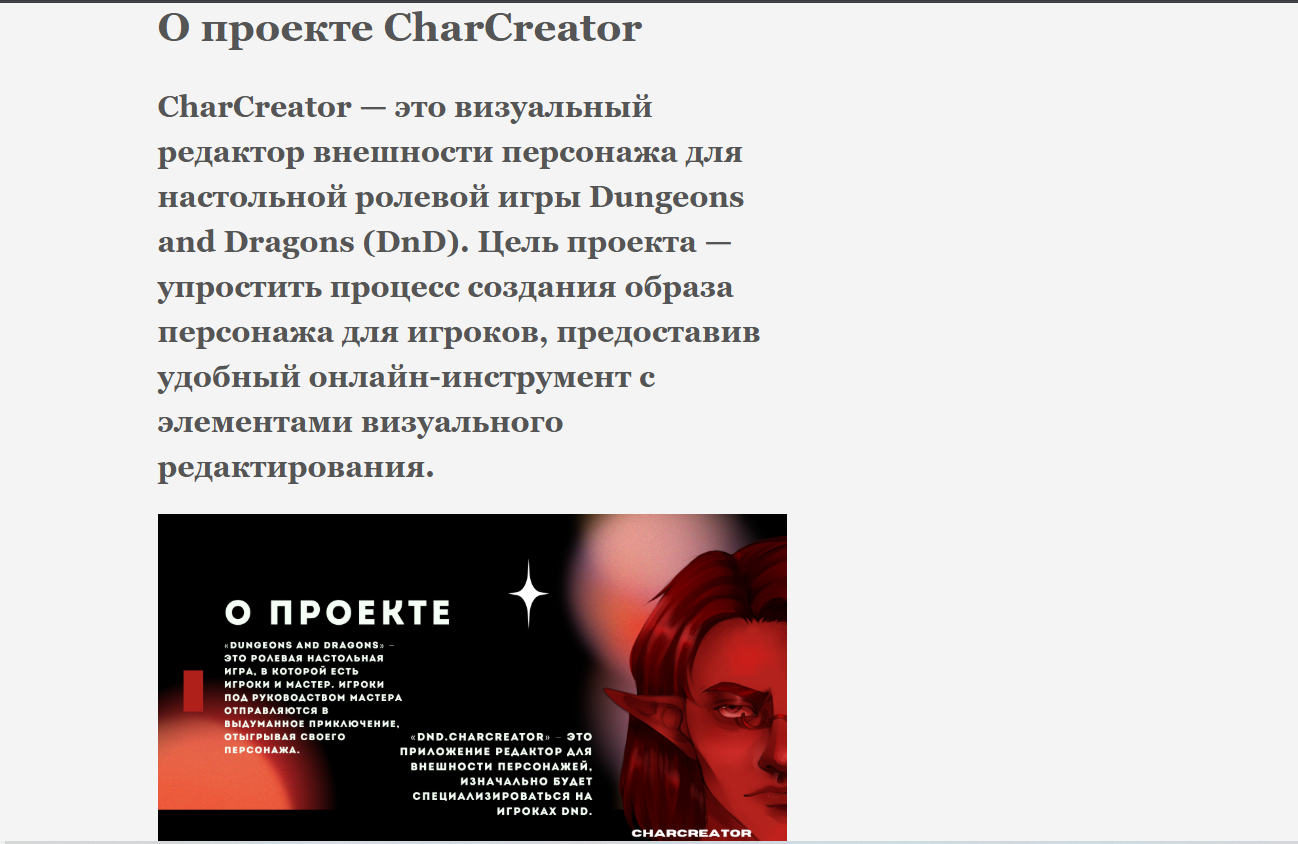


Рисунок 4 - страница about 2

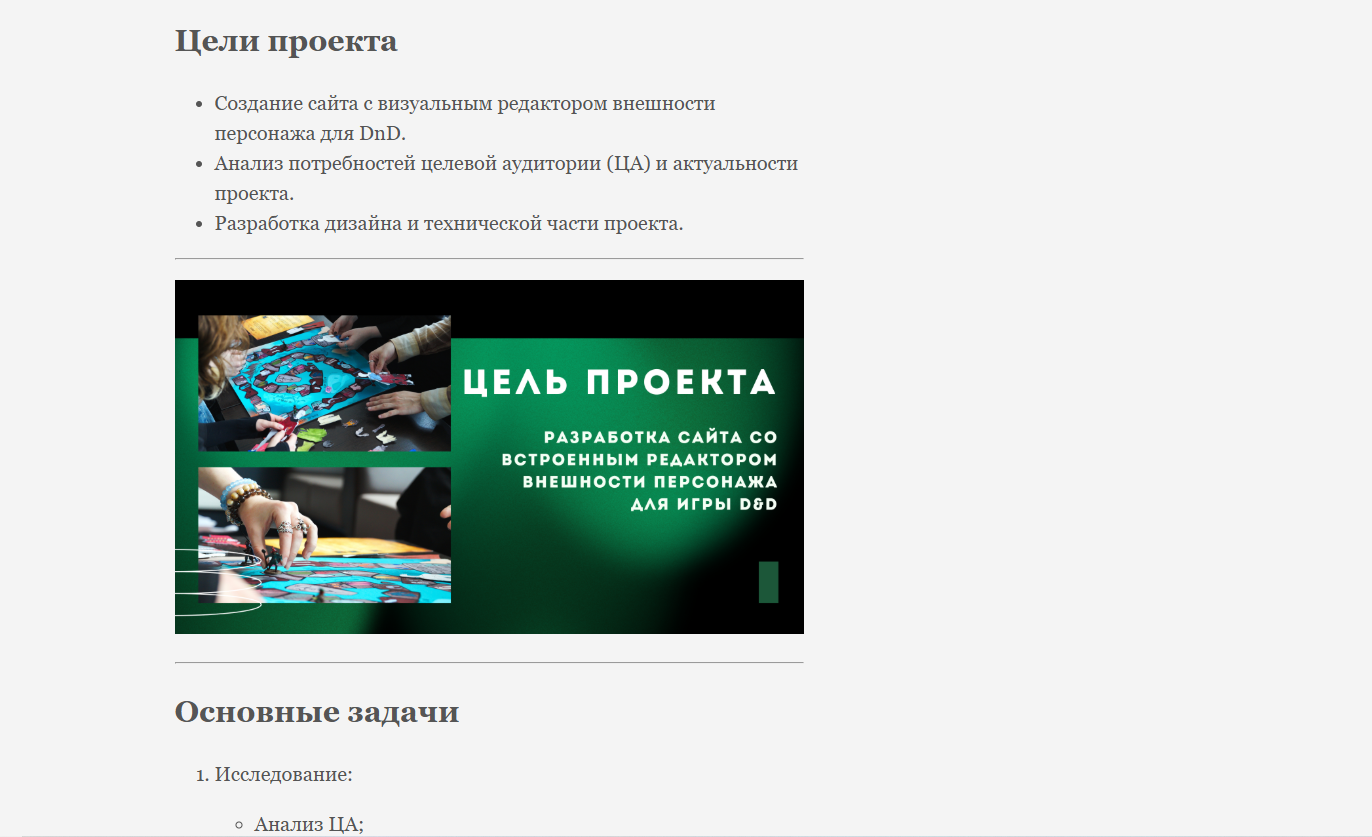


Рисунок 5 - страница about 3

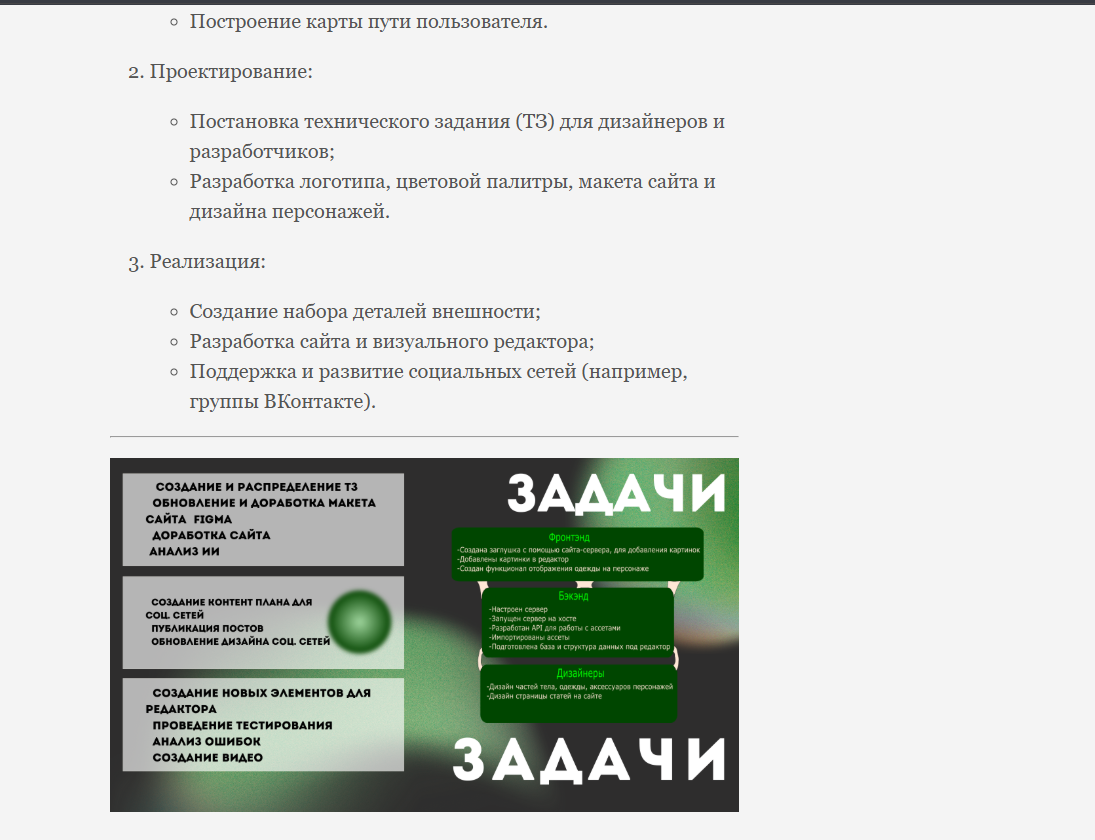


Рисунок 6 - страница about 4

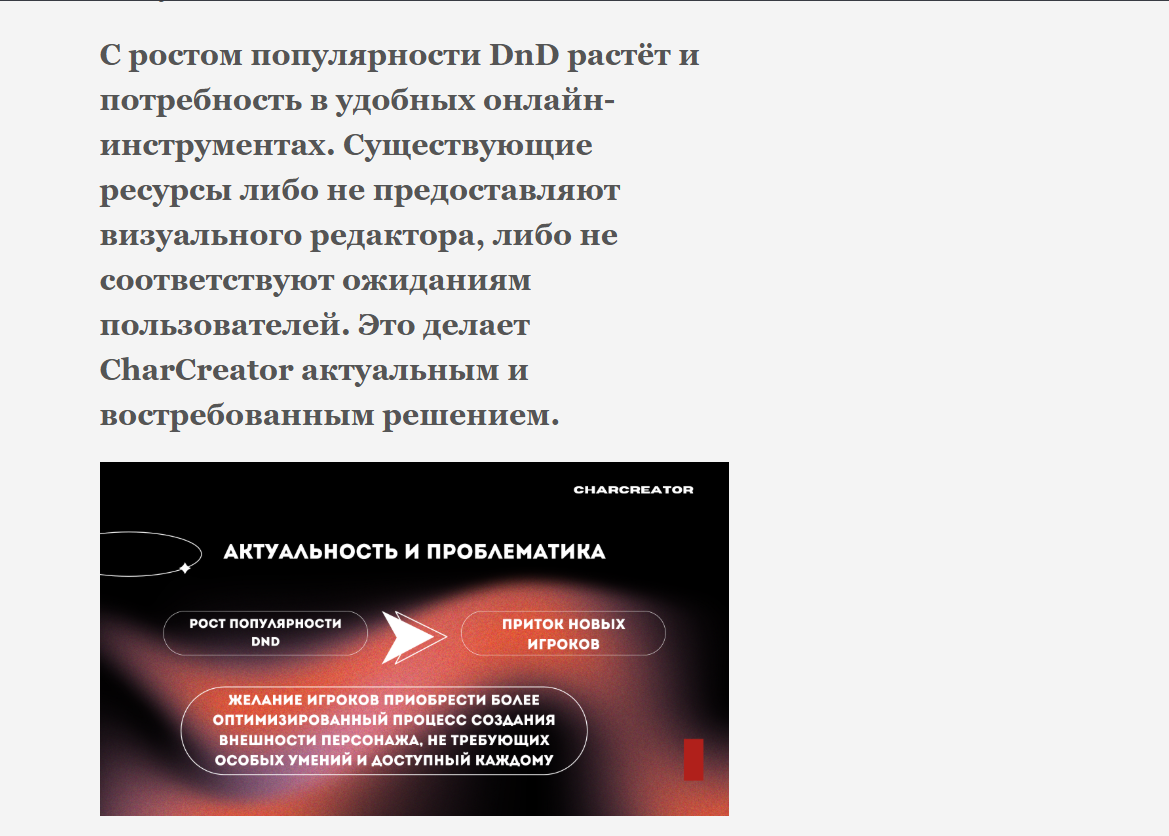


Рисунок 7 - страница about 5

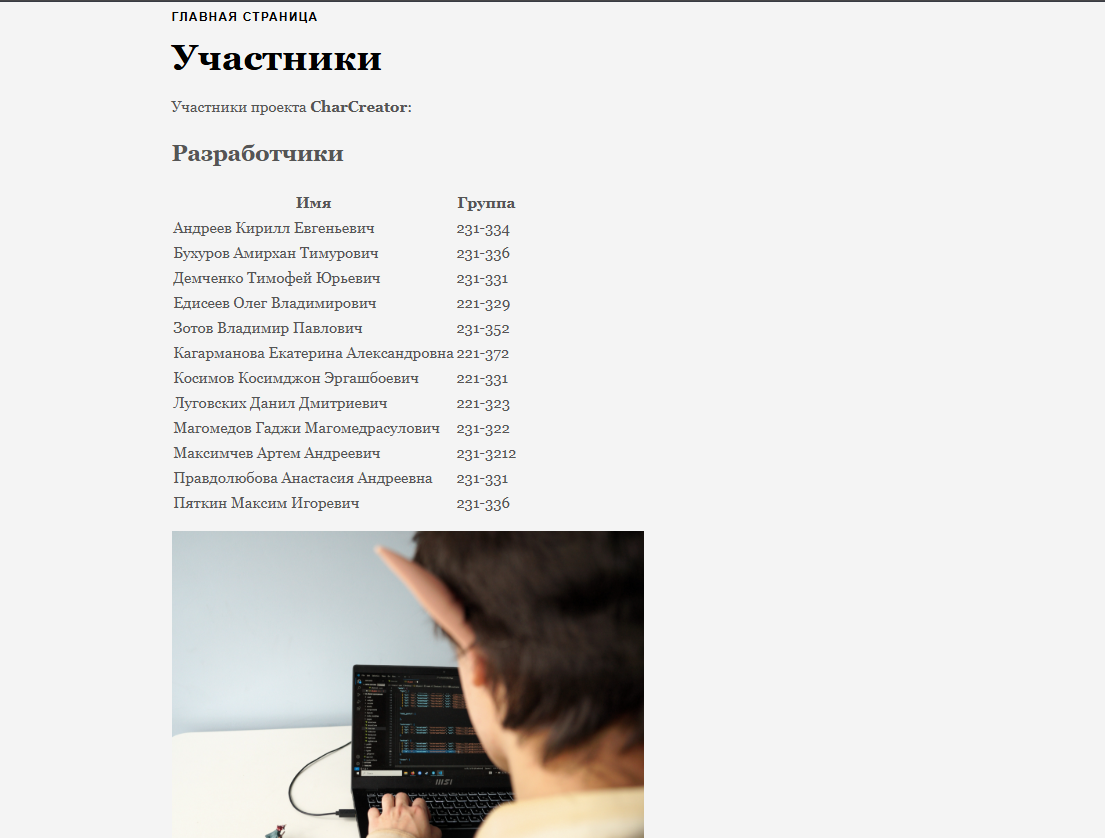


Рисунок 8 - страница team 1

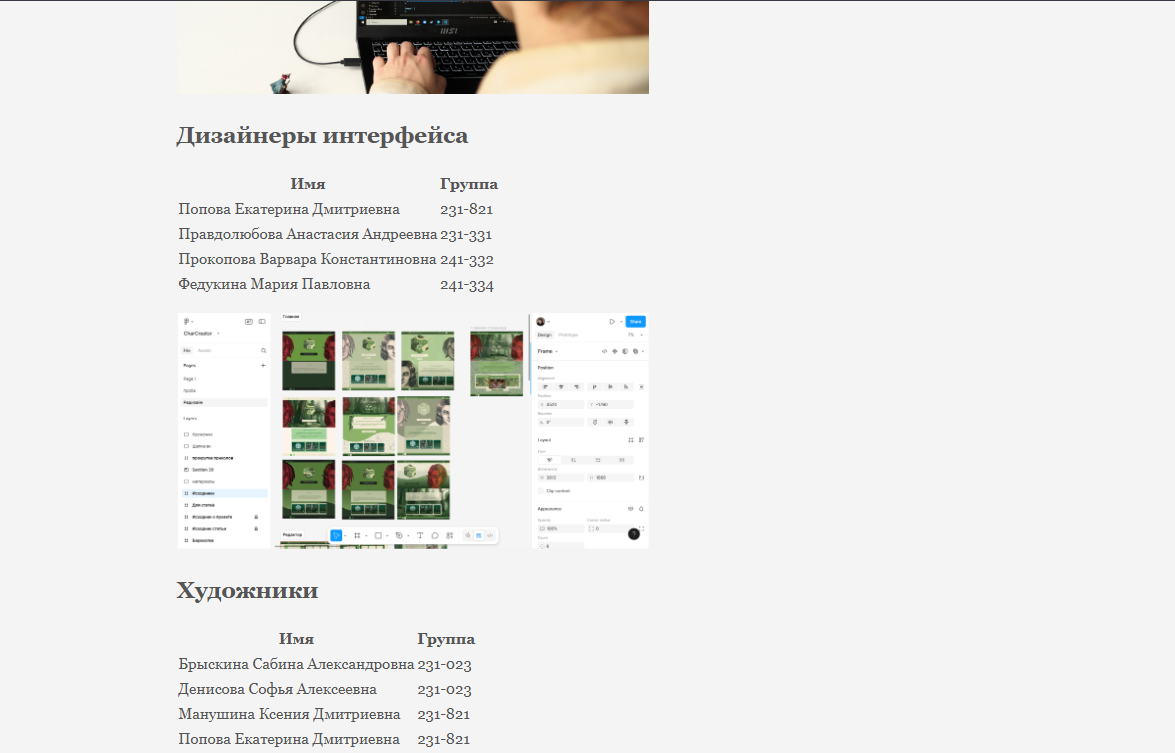


Рисунок 9 - страница team 2

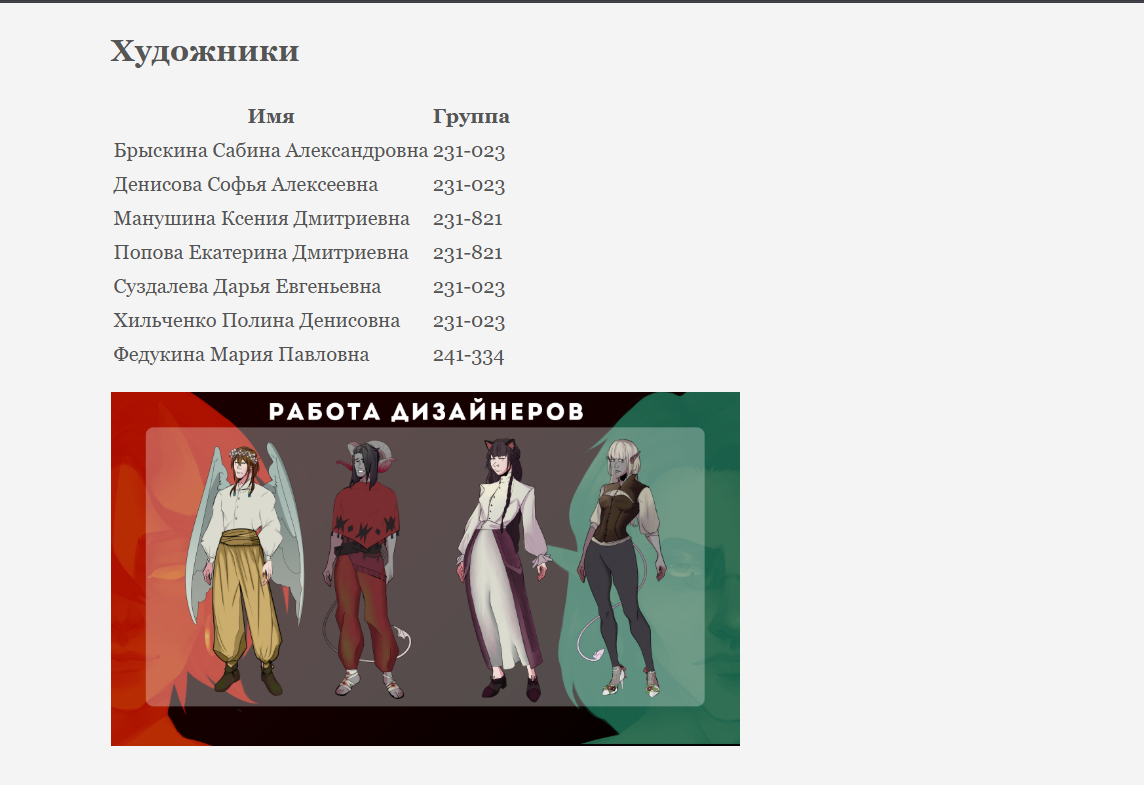


Рисунок 10 - страница team 3

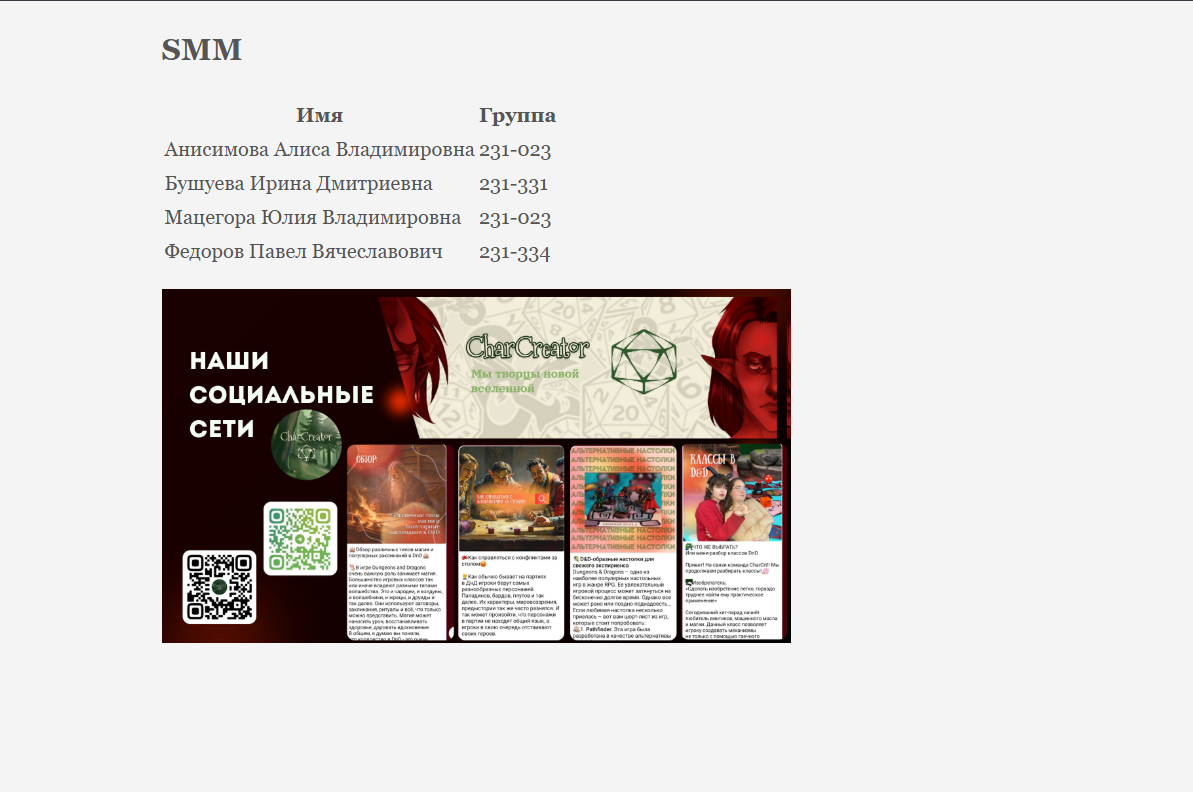


Рисунок 11 - страница team 4

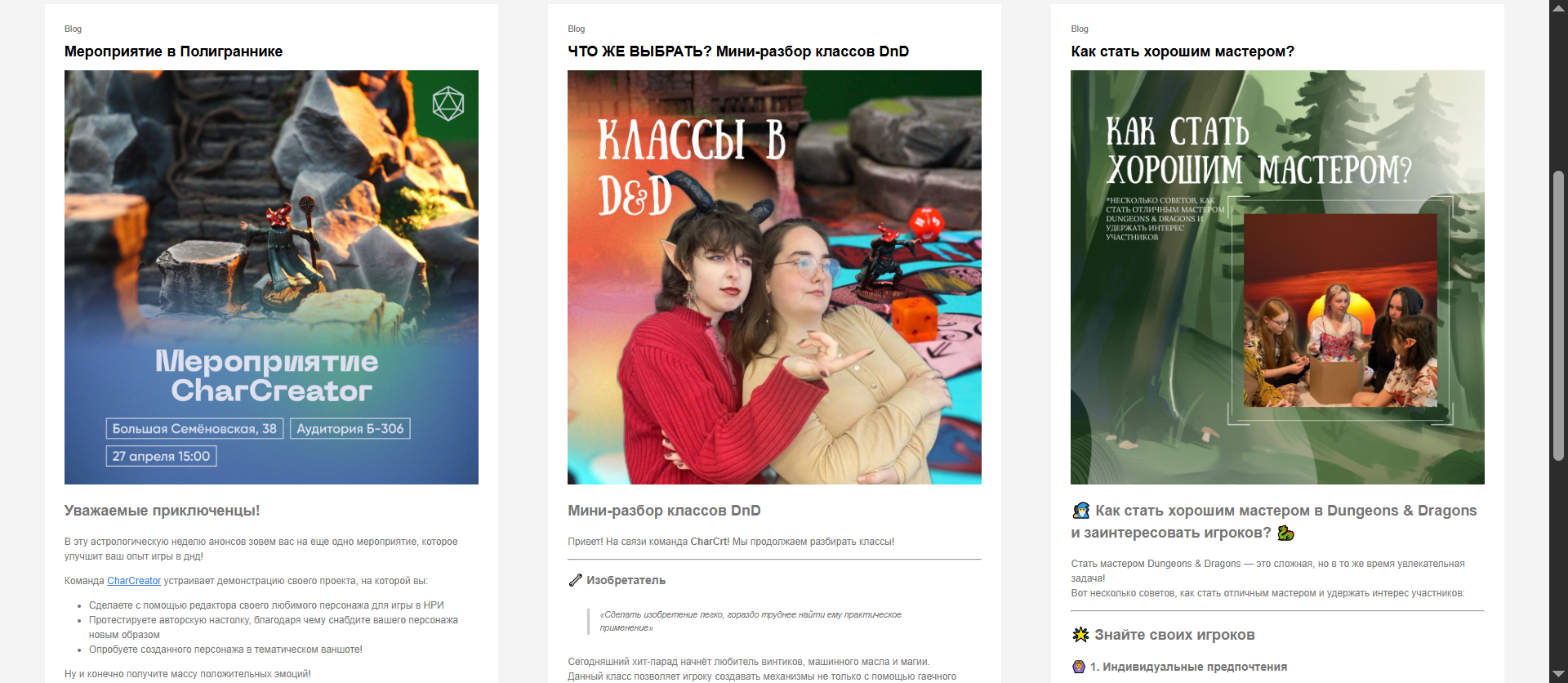


Рисунок 12 - страница blog 1

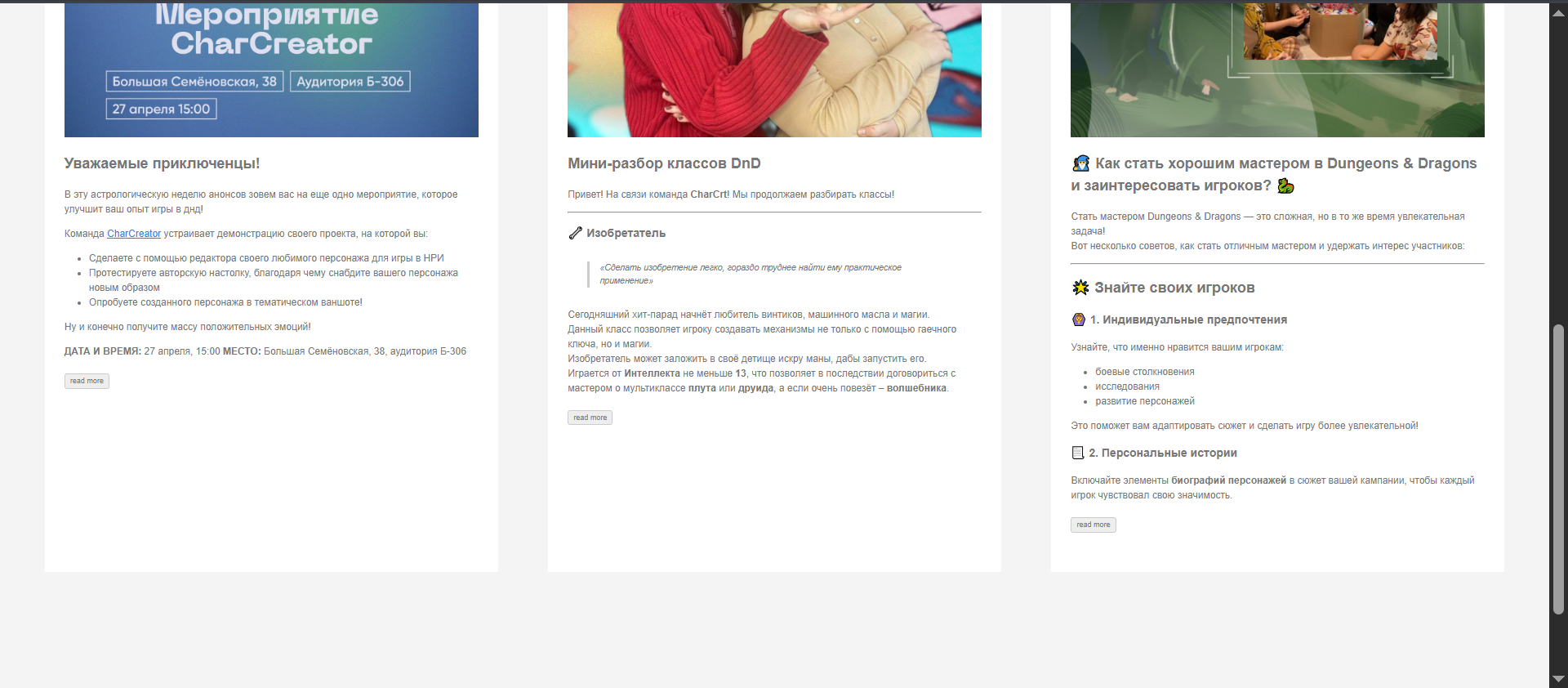


Рисунок 13 - страница blog 2

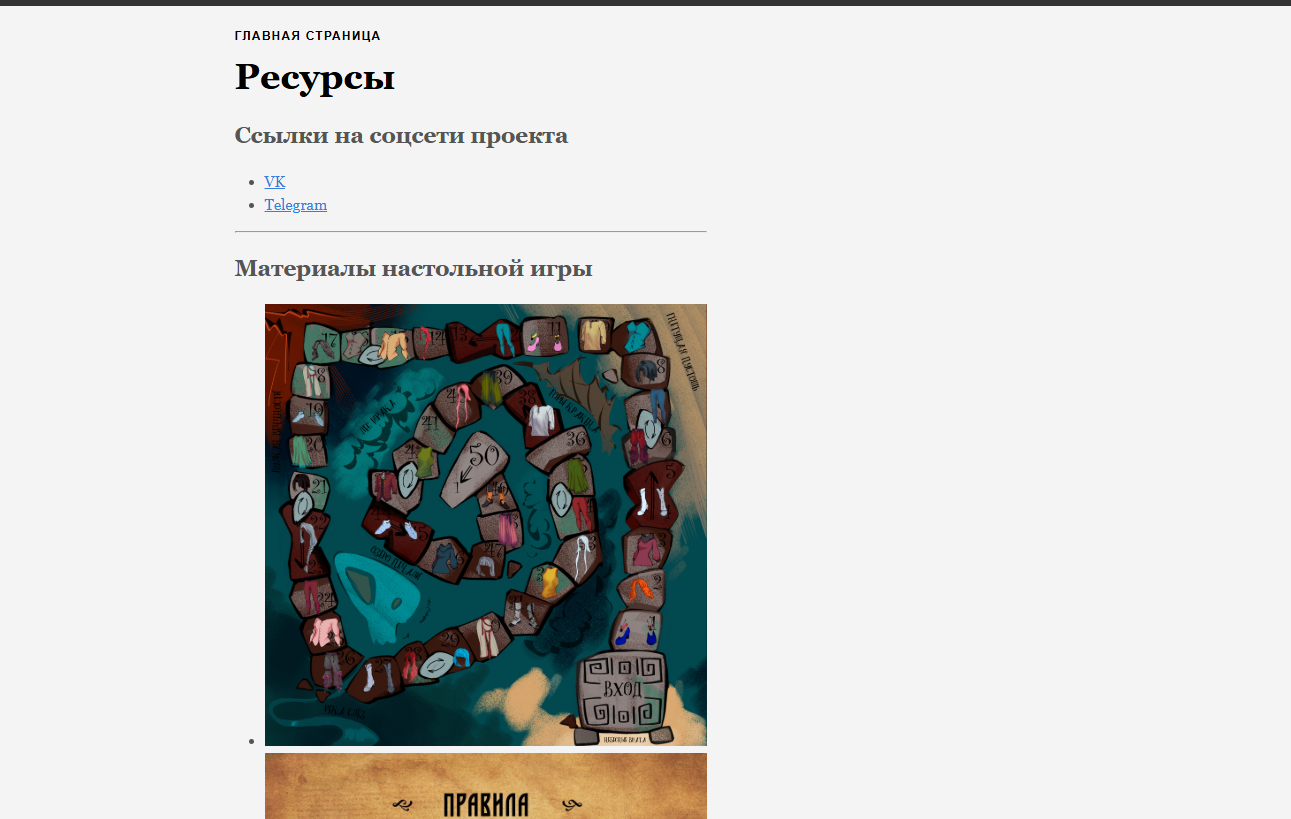


Рисунок 14 - страница resources 1

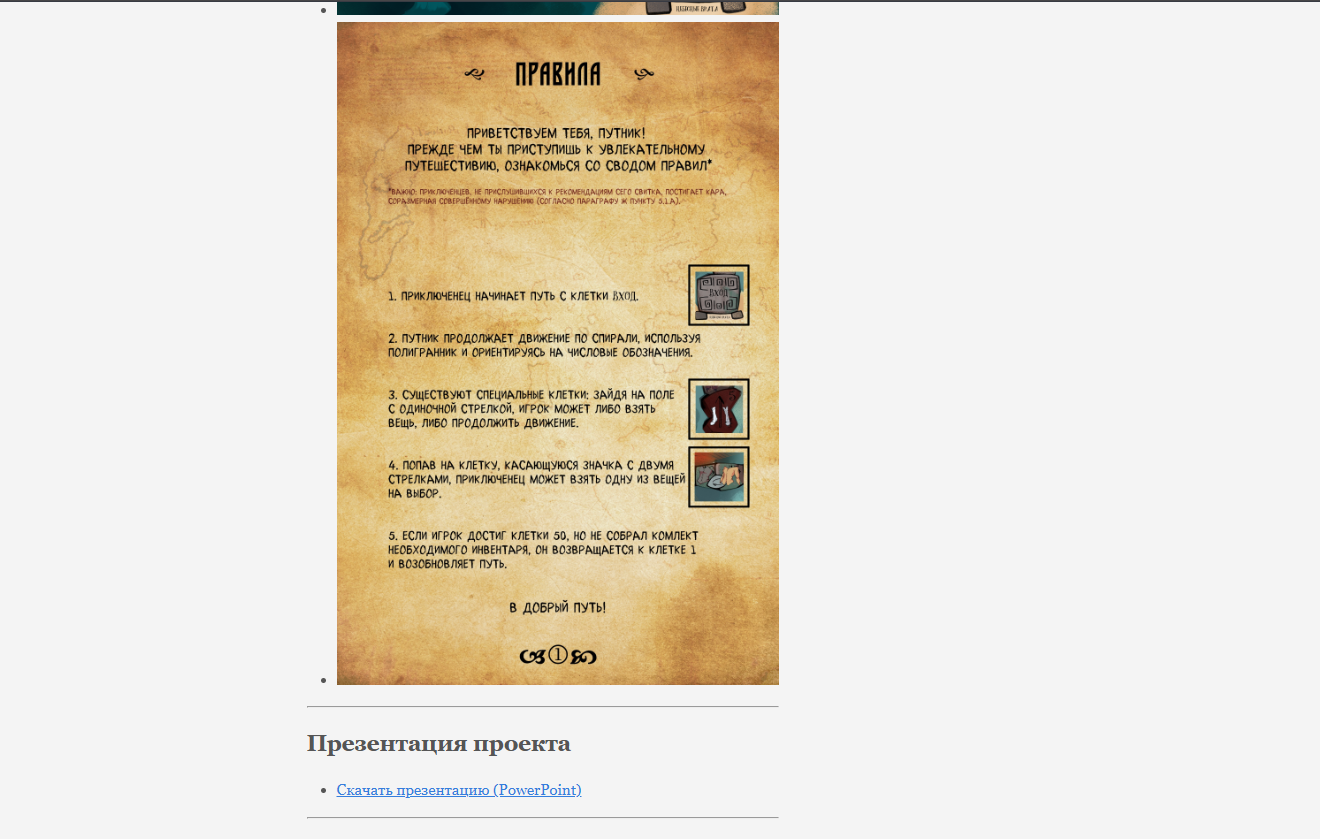


Рисунок 15 - страница resources 2

import numpy as np  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import accuracy\_score, confusion\_matrix  
import matplotlib.pyplot as plt  
import random  
  
#################################### ДАННЫЕ ##############################################  
my\_moves = ['Б', 'К', 'Б', 'Б', 'Б', 'Н', 'Н', 'К', 'Б', 'Б',  
 'Н', 'Б', 'Б', 'Б', 'Н', 'К', 'Б', 'К', 'Б', 'Н',  
 'Б', 'К', 'Н', 'Б', 'Н', 'К', 'Б', 'Б', 'Н', 'К',  
 'К', 'Н', 'Б', 'Б', 'Н', 'К', 'Н', 'К', 'Б', 'Н',  
 'Н', 'К', 'Н', 'Б', 'К', 'Н', 'Б', 'Б', 'К', 'Н',  
 'Н', 'К', 'Б', 'Б', 'Б', 'К', 'Н', 'Б', 'Б', 'Н',  
 'К', 'Б', 'К', 'Б', 'К', 'К', 'К', 'Б', 'К', 'Б',  
 'К', 'Н', 'К', 'К', 'Н', 'Б', 'Б', 'Б', 'Б', 'Б',  
 'Н', 'Н', 'Б', 'К', 'К', 'Б', 'Б', 'К', 'Н', 'Б',  
 'Б', 'Н', 'Б', 'Н', 'К', 'Н', 'Н', 'К', 'Б', 'К',  
 'Б', 'К', 'Б', 'Б', 'К', 'Н', 'Н', 'К', 'Б', 'Н',  
 'К', 'К', 'Н', 'Б', 'Б', 'Б', 'К', 'К', 'Н', 'Б',  
 'К', 'Б', 'Н', 'Б', 'Б', 'К', 'К', 'Н', 'К', 'Н',  
 'Б', 'К', 'Б', 'Б', 'Н', 'К', 'К', 'Б', 'Б', 'К',  
 'Н', 'К', 'Н', 'К', 'Б', 'Н', 'Б', 'К', 'К', 'Б',  
 'К', 'Б', 'К', 'Н', 'Н', 'Б', 'Н', 'К', 'Н', 'К',  
 'Б', 'Н', 'Б', 'К', 'Б', 'К', 'Б', 'Н', 'К', 'Б',  
 'К', 'К', 'Н', 'Б', 'К', 'Н', 'Н', 'К', 'Б', 'Б',  
 'Б', 'Н', 'Н', 'Б', 'К', 'К', 'Н', 'Б', 'Б', 'К',  
 'К', 'Б', 'Н', 'К', 'Б', 'Н', 'К', 'Б', 'К', 'Б']  
  
opponent\_moves = ['Н', 'Н', 'К', 'К', 'Н', 'Н', 'Н', 'Н', 'Н', 'К',  
 'К', 'К', 'К', 'Н', 'Б', 'Н', 'Н', 'Н', 'Б', 'Б',  
 'Н', 'К', 'К', 'Б', 'Н', 'Б', 'Б', 'Н', 'Б', 'К',  
 'Б', 'К', 'К', 'Б', 'Н', 'Б', 'К', 'Б', 'Б', 'Н',  
 'К', 'Б', 'Н', 'Б', 'К', 'Б', 'Б', 'Н', 'Б', 'Б',  
 'Н', 'Н', 'Б', 'Н', 'Н', 'К', 'К', 'Б', 'Н', 'К',  
 'Н', 'Н', 'Б', 'Н', 'К', 'К', 'Н', 'Н', 'К', 'Б',  
 'К', 'Н', 'К', 'Н', 'Б', 'Н', 'К', 'К', 'К', 'Н',  
 'Б', 'Б', 'К', 'К', 'Н', 'Н', 'К', 'К', 'Н', 'К',  
 'Н', 'Б', 'Н', 'Б', 'К', 'К', 'Б', 'Б', 'Н', 'Н',  
 'Н', 'К', 'К', 'Б', 'Н', 'Н', 'Б', 'К', 'К', 'Б',  
 'Б', 'Н', 'Б', 'К', 'Н', 'К', 'Н', 'Н', 'Б', 'Б',  
 'К', 'К', 'Н', 'Н', 'Б', 'Н', 'К', 'К', 'Б', 'К',  
 'Н', 'Б', 'К', 'Н', 'К', 'К', 'Н', 'Н', 'Б', 'Б',  
 'К', 'Н', 'Н', 'Б', 'Н', 'К', 'Н', 'К', 'Б', 'К',  
 'К', 'Б', 'К', 'Н', 'Н', 'К', 'Н', 'Б', 'К', 'Н',  
 'Б', 'К', 'Б', 'К', 'Н', 'Б', 'К', 'Н', 'К', 'К',  
 'Б', 'К', 'Н', 'Н', 'К', 'Б', 'К', 'Н', 'Б', 'Н',  
 'К', 'Б', 'Б', 'Н', 'К', 'К', 'Н', 'К', 'Б', 'Н',  
 'К', 'Б', 'Н', 'Б', 'К', 'Б', 'К', 'Н', 'Н', 'Б']  
  
  
def generate\_smart\_sequence(length=100):  
 moves = ['R', 'P', 'S']  
 sequence = []  
  
 for i in range(length):  
 if len(sequence) >= 3 and sequence[-1] == sequence[-2] == sequence[-3]:  
 choices = [m for m in moves if m != sequence[-1]]  
 next\_move = random.choice(choices)  
 elif len(sequence) >= 2 and sequence[-1] == sequence[-2]:  
 if random.random() < 0.7:  
 choices = [m for m in moves if m != sequence[-1]]  
 next\_move = random.choice(choices)  
 else:  
 next\_move = sequence[-1]  
 else:  
 # В остальных случаях — 60% сменить  
 if len(sequence) > 0 and random.random() < 0.6:  
 choices = [m for m in moves if m != sequence[-1]]  
 next\_move = random.choice(choices)  
 else:  
 next\_move = random.choice(moves)  
  
 sequence.append(next\_move)  
  
 return sequence  
  
smart\_my\_moves = generate\_smart\_sequence(500)  
smart\_opponent\_moves = generate\_smart\_sequence(500)  
  
rps\_to\_russian = {'R': 'К', 'P': 'Б', 'S': 'Н'}  
smart\_my\_moves = [rps\_to\_russian[m] for m in smart\_my\_moves]  
smart\_opponent\_moves = [rps\_to\_russian[m] for m in smart\_opponent\_moves]  
  
my\_moves += smart\_my\_moves  
opponent\_moves += smart\_opponent\_moves  
  
move\_to\_int = {'К': 0, 'Н': 1, 'Б': 2}  
int\_to\_move = {0: 'Камень', 1: 'Ножницы', 2: 'Бумага'}  
  
X = [] # история ходов игрока + противника  
y = [] # следующий ход противника  
  
history\_size = 5 # сколько прошлых ходов учитывать  
  
for i in range(history\_size, len(my\_moves)):  
 my\_history = my\_moves[i - history\_size:i]  
 opp\_history = opponent\_moves[i - history\_size:i]  
 encoded = []  
  
 for m, o in zip(my\_history, opp\_history):  
 # one-hot игрока  
 onehot\_my = [0, 0, 0]  
 onehot\_my[move\_to\_int[m]] = 1  
 encoded.extend(onehot\_my)  
  
 # one-hot противника  
 onehot\_opp = [0, 0, 0]  
 onehot\_opp[move\_to\_int[o]] = 1  
 encoded.extend(onehot\_opp)  
  
 X.append(encoded)  
 y.append(move\_to\_int[opponent\_moves[i]])  
  
X = np.array(X)  
y = np.array(y)  
y\_onehot = np.eye(3)[y]  
  
######################################## архитектура сети ################################################  
  
def sigmoid(x, deriv=False):  
 x = np.clip(x, -500, 500)  
 if deriv:  
 return x \* (1 - x)  
 return 1 / (1 + np.exp(-x))  
  
  
def softmax(x):  
 exp\_x = np.exp(x - np.max(x, axis=1, keepdims=True))  
 return exp\_x / np.sum(exp\_x, axis=1, keepdims=True)  
  
def predict(X):  
 l1 = sigmoid(np.dot(X, syn0))  
 l2 = softmax(np.dot(l1, syn1))  
 return np.argmax(l2, axis=1)  
  
input\_size = history\_size \* 3 \* 2 # 3 значения (К,Н,Б) \* 2 игрока \* history\_size  
hidden\_size = 16 # нейронов в скрытом слое  
output\_size = 3 # 3 возможных хода (К, Н, Б)  
  
np.random.seed(42)  
syn0 = 2 \* np.random.random((input\_size, hidden\_size)) - 1 # веса вход -> скрытый слой  
syn1 = 2 \* np.random.random((hidden\_size, output\_size)) - 1 # веса скрытый -> выходной слой  
  
learning\_rate = 0.01 # скорость обучения  
epochs = 5000 # количество итераций обучения  
l2\_lambda = 0.01  
  
####################################### обучение сети ########################################  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y\_onehot, test\_size=0.2, random\_state=42)  
  
loss\_history = []  
  
for epoch in range(epochs):  
 l0 = X\_train  
 l1 = sigmoid(np.dot(l0, syn0))  
 l2 = softmax(np.dot(l1, syn1))  
  
 l2\_error = y\_train - l2  
 l2\_delta = l2\_error # для softmax + cross-entropy  
  
 l1\_error = l2\_delta.dot(syn1.T)  
 l1\_delta = l1\_error \* sigmoid(l1, deriv=True)  
  
 syn1 += (l1.T.dot(l2\_delta) - l2\_lambda \* syn1) \* learning\_rate  
 syn0 += (l0.T.dot(l1\_delta) - l2\_lambda \* syn0) \* learning\_rate  
  
 loss = np.mean(np.abs(l2\_error))  
 loss\_history.append(loss)  
  
 if epoch % 1000 == 0:  
 print(f"Epoch {epoch}, Loss: {loss:.4f}")  
  
###### оценка модели #######  
def evaluate\_model(X, y\_true\_onehot):  
 y\_pred = predict(X)  
 y\_true = np.argmax(y\_true\_onehot, axis=1)  
 accuracy = accuracy\_score(y\_true, y\_pred)  
 cm = confusion\_matrix(y\_true, y\_pred)  
 return accuracy, cm  
  
train\_accuracy, train\_cm = evaluate\_model(X\_train, y\_train)  
test\_accuracy, test\_cm = evaluate\_model(X\_test, y\_test)  
  
print(f"\nTrain Accuracy: {train\_accuracy:.2%}")  
print(f"Test Accuracy: {test\_accuracy:.2%}")  
  
plt.plot(loss\_history)  
plt.title("График ошибки при обучении")  
plt.xlabel("Epoch")  
plt.ylabel("Loss")  
plt.show()  
  
  
####################################### ИНТЕРАКТИВНОЕ ДЕМО ########################################  
  
# def play\_vs\_ai():  
# print("\nДемо-режим: игра против ИИ")  
# print("Вводите свои ходы (К - камень, Н - ножницы, Б - бумага, Q - выход)")  
#  
# # история последних ходов (игрока и ИИ)  
# game\_history = []  
#  
# while True:  
# # Ввод хода игрока  
# player\_move = input("\nВаш ход (К/Н/Б/Q): ").upper()  
# if player\_move == 'Q':  
# print("Игра завершена.")  
# break  
# if player\_move not in ['К', 'Н', 'Б']:  
# print("Некорректный ввод! Используйте К, Н или Б.")  
# continue  
#  
# # если накопилось достаточно истории, делаем предсказание  
# ai\_move = None  
# if len(game\_history) >= history\_size:  
# encoded = []  
# for m, o in game\_history[-history\_size:]:  
# onehot\_my = [0, 0, 0]  
# onehot\_my[move\_to\_int[m]] = 1  
# encoded.extend(onehot\_my)  
#  
# onehot\_opp = [0, 0, 0]  
# onehot\_opp[move\_to\_int[o]] = 1  
# encoded.extend(onehot\_my)  
#  
# prediction = predict(np.array([encoded]))[0]  
# ai\_move = int\_to\_move[prediction]  
# else:  
# # если истории мало случайный ход  
# ai\_move = random.choice(['К', 'Н', 'Б'])  
#  
# result = "Ничья"  
# if (player\_move == 'К' and ai\_move == 'Н') or \  
# (player\_move == 'Н' and ai\_move == 'Б') or \  
# (player\_move == 'Б' and ai\_move == 'К'):  
# result = "Вы победили!"  
# elif (ai\_move == 'К' and player\_move == 'Н') or \  
# (ai\_move == 'Н' and player\_move == 'Б') or \  
# (ai\_move == 'Б' and player\_move == 'К'):  
# result = "ИИ победил!"  
#  
# game\_history.append((player\_move, ai\_move))  
#  
# print(f"\nИИ выбрал: {ai\_move}")  
# print(f"Результат: {result}")  
# print(f"История (последние {min(3, len(game\_history))} раунда):")  
# for i, (p, a) in enumerate(game\_history[-3:]):  
# print(f"Раунд {len(game\_history) - 3 + i + 1}: Вы - {p}, ИИ - {a}")  
# play\_vs\_ai()

Приложение 16 – листинг нейросети

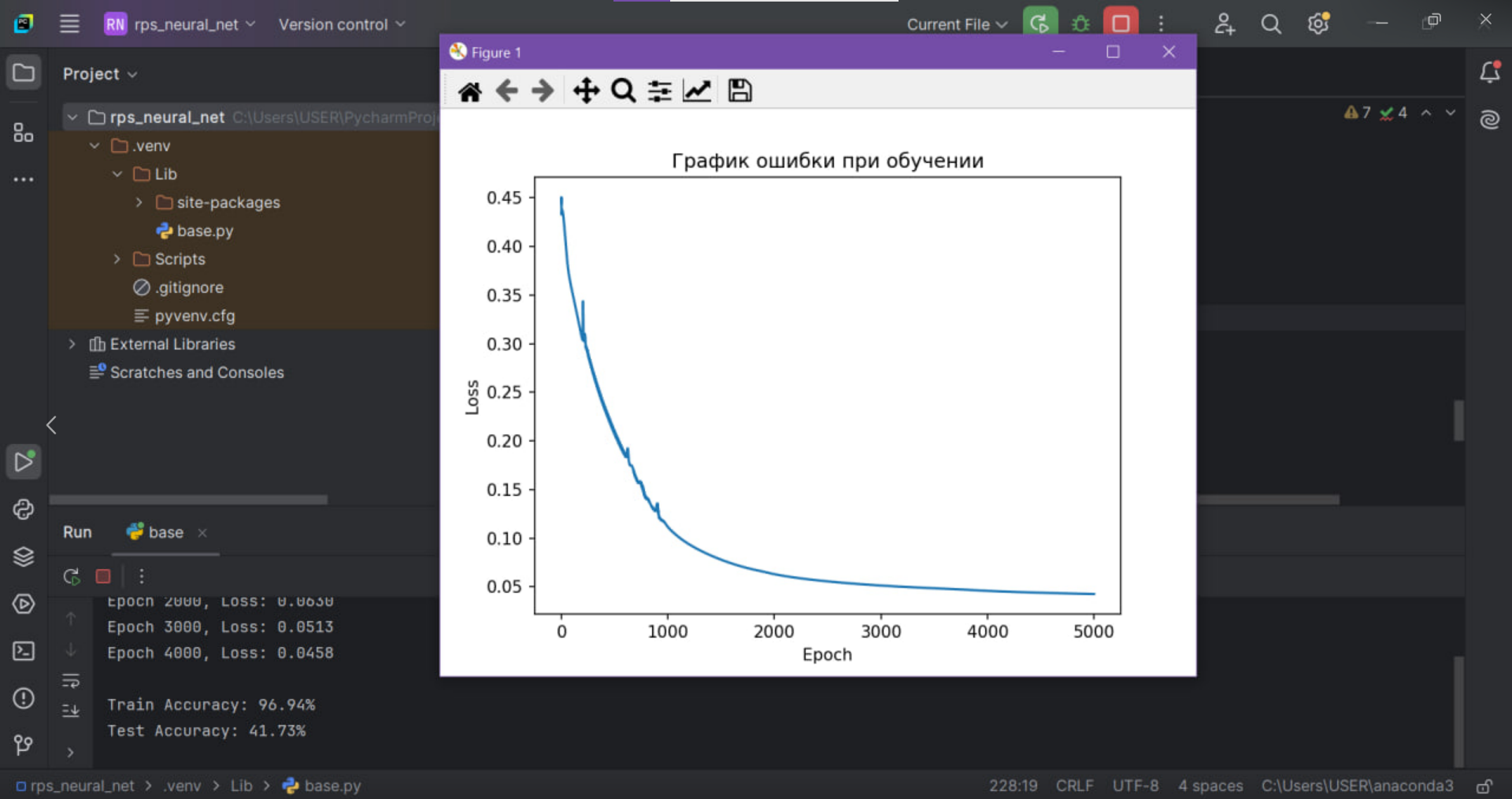


Рисунок 16 - вывод при запуске