Федеральное государственное автономное образовательное учреждение высшего образования

«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий  
Кафедра «Инфокогнитивные технологии»

Направление подготовки/ специальность: 09.03.01 Информатика и вычислительная техника/ Системная и программная инженерия

ОТЧЕТ

по проектной практике

Студент: Клевин Александр Ильич Группа: 241-3211

Место прохождения практики: Московский Политехнический Университет, кафедра «Инфокогнитивные технологии»

Отчет принят с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва 2025

# **ОГЛАВЛЕНИЕ**

[**ОГЛАВЛЕНИЕ** 2](#_Toc199702769)

[**ВВЕДЕНИЕ** 3](#_Toc199702770)

[**1.** **ОБЩАЯ ИНФОРМАЦИЯ О ПРОЕКТЕ** 4](#_Toc199702771)

[1.1 Название проекта 4](#_Toc199702772)

[1.2 Цели и задачи проекта 4](#_Toc199702773)

[**2.** **ОБЩАЯ ХАРАКТЕРИСТИКА ДЕЯТЕЛЬНОСТИ ОРГАНИЗАЦИИ** 5](#_Toc199702774)

[2.1 Наименование партнера 5](#_Toc199702775)

[**3.** **ОПИСАНИЕ ЗАДАНИЯ ПО ПРОЕКТНОЙ ПРАКТИКЕ** 6](#_Toc199702776)

[3.1 Базовая часть 6](#_Toc199702777)

[3.2 Вариативная часть 6](#_Toc199702778)

[**4.** **ОПИСАНИЕ ДОСТИГНУТЫХ РЕЗУЛЬТАТОВ ПО ПРОЕКТНОЙ ПРАКТИКЕ** 7](#_Toc199702779)

[4.1 Базовая часть 7](#_Toc199702780)

[4.1.1 Экскурсия в компанию "Первый Бит" 10](#_Toc199702781)

[5.2 Вариативная часть 12](#_Toc199702782)

[5.2.1 Описание задачи проекта 12](#_Toc199702783)

[5.2.2 Пошаговое руководство по созданию 12](#_Toc199702784)

[**ЗАКЛЮЧЕНИЕ** 17](#_Toc199702785)

[**CПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ** 18](#_Toc199702786)

# **ВВЕДЕНИЕ**

В данном отчёте представлены результаты моей учебной практики. В ходе работы над проектом я применил теоретические знания, полученные в процессе обучения, а также приобрёл новые профессиональные навыки в области информационных технологий и разработки игр. Работа над проектом осуществлялась в рамках выданного задания, включающего разработку игры с использованием актуальных технологий и инструментов.

**Основные цели практики:**

* освоить работу с системами контроля версий, в частности с Git;
* получить практический опыт в создании статических веб-страниц с использованием HTML и CSS;
* изучить методы оформления проектной документации с помощью языка разметки Markdown;
* развить навыки эффективного взаимодействия в команде.

# **ОБЩАЯ ИНФОРМАЦИЯ О ПРОЕКТЕ**

1.1 Название проекта

В рамках проектной практики я участвую в разработке игры под названием **Компьютерная игра для одного человека «ECOllapse»**, которая представляет собой симулятор выживания с динамической экосистемой и процедурной генерацией мира.

1.2 Цели и задачи проекта

**Цель проекта** - Создать компьютерную игру с процедурно генерируемыми биомами, динамической экосистемой и механиками выживания, которая будет обеспечивать игроку уникальный опыт и высокую реиграбельность.

**Задачи проекта:**

* Дизайн концепции;
* Базовый прототип;
* Гибкая система Эволюции;
* Симуляция экосистем;
* Уникальные условия;
* Тестирование и балансировка;
* Разработка финальной версии.

# **ОБЩАЯ ХАРАКТЕРИСТИКА ДЕЯТЕЛЬНОСТИ ОРГАНИЗАЦИИ**

## 2.1 Наименование партнера

Партнером проекта является Московский Политехнический Университет.

# **ОПИСАНИЕ ЗАДАНИЯ ПО ПРОЕКТНОЙ ПРАКТИКЕ**

## 3.1 Базовая часть

В рамках базовой части проектной практики передо мной стояли следующие задачи, представленные в файле <https://github.com/mospol/practice-2025-1/tree/master/task> :

* создать персональный репозиторий на GitHub, освоить команды git;
* изучить синтаксис Markdown и подготовить документацию;
* разработать статический веб-сайт с использованием HTML и CSS;
* провзаимодействовать с партнерами;
* подготовить отчёт по практике в форматах DOCX и PDF.

## 

## 3.2 Вариативная часть

В рамках проектной практики я разработал телеграм-бота-помощника для игры «ECOllabse: Миры бесконечной адаптации».

Выполнил:

Клевин Александр Ильич 241-3211.

# **ОПИСАНИЕ ДОСТИГНУТЫХ РЕЗУЛЬТАТОВ ПО ПРОЕКТНОЙ ПРАКТИКЕ**

## 4.1 Базовая часть

В ходе выполнения базовой части практики были достигнуты следующие результаты:

1. Создан и настроен Git-репозиторий. (https://github.com/kkrraauuzzee/practice.git).

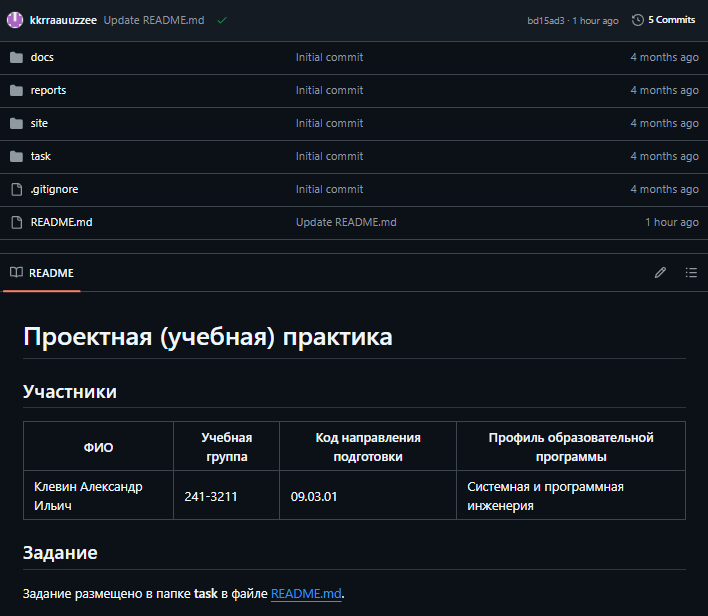


Рисунок 1. Скриншот репозитория

1. Подготовлена документация в формате Markdown.
2. Разработан статический сайт на HTML, описывающий проект по дисциплине «Проектная деятельность». Выполнены условия с обязательными страницами.

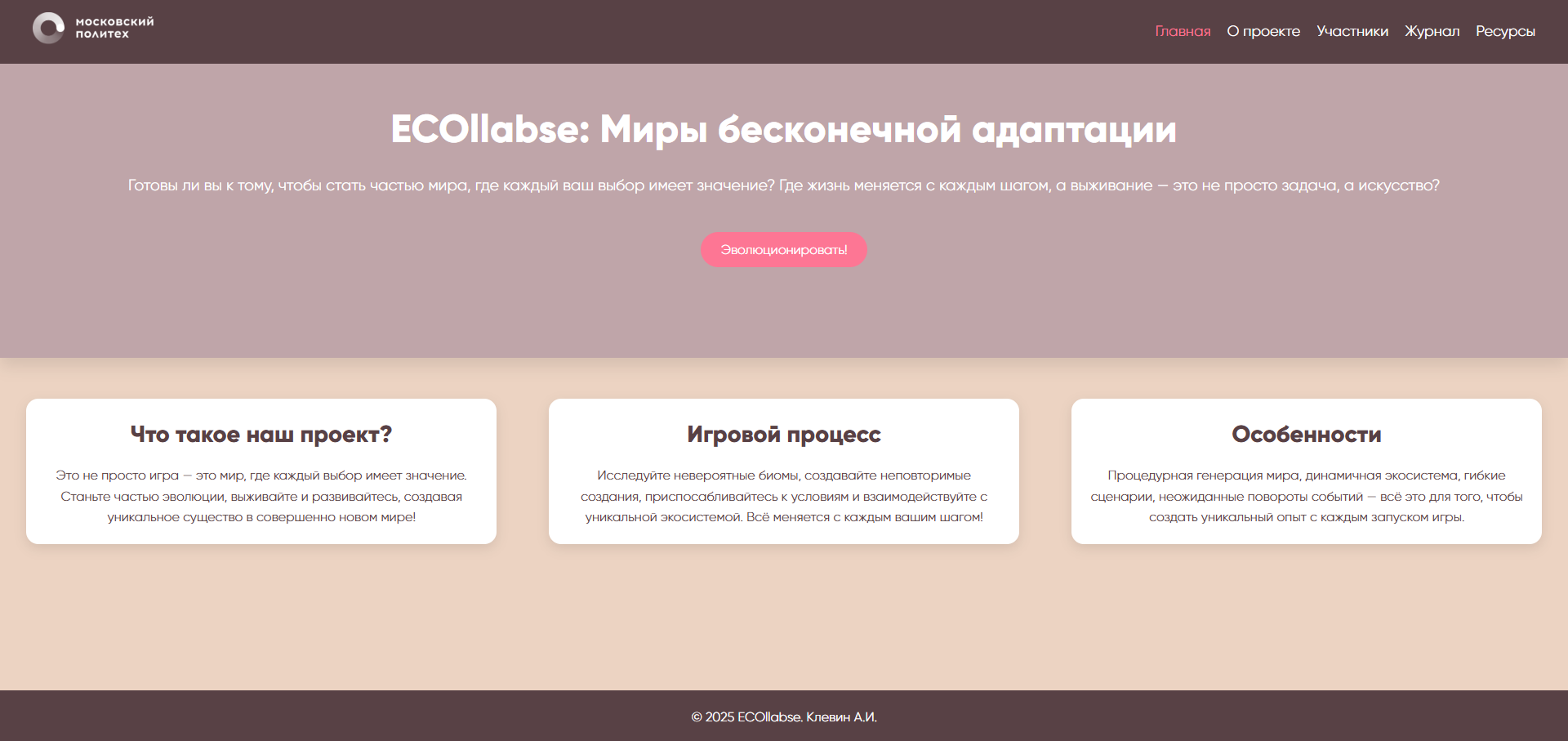


Рисунок 2. Страница «Главная»

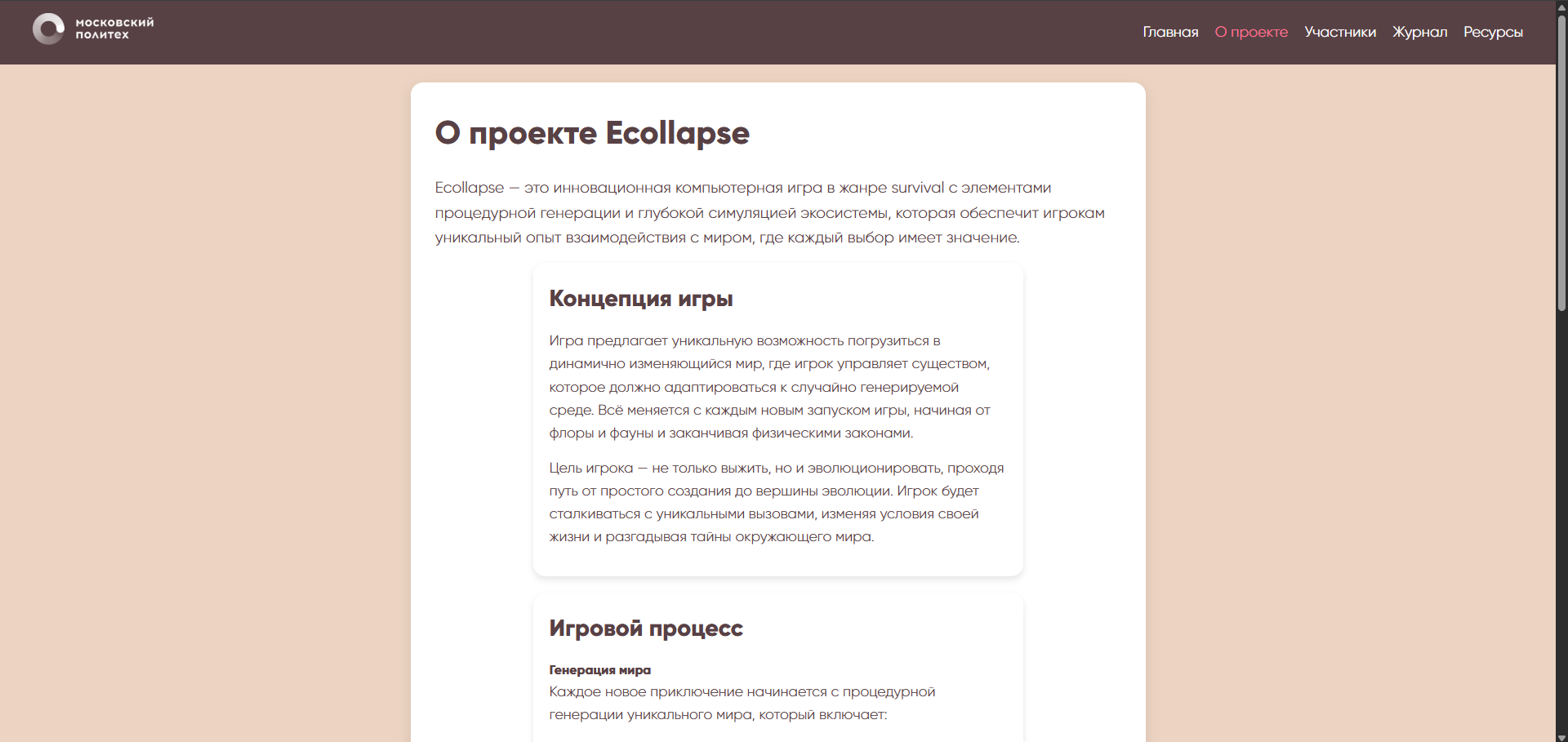


Рисунок 3. Страница «О проекте»

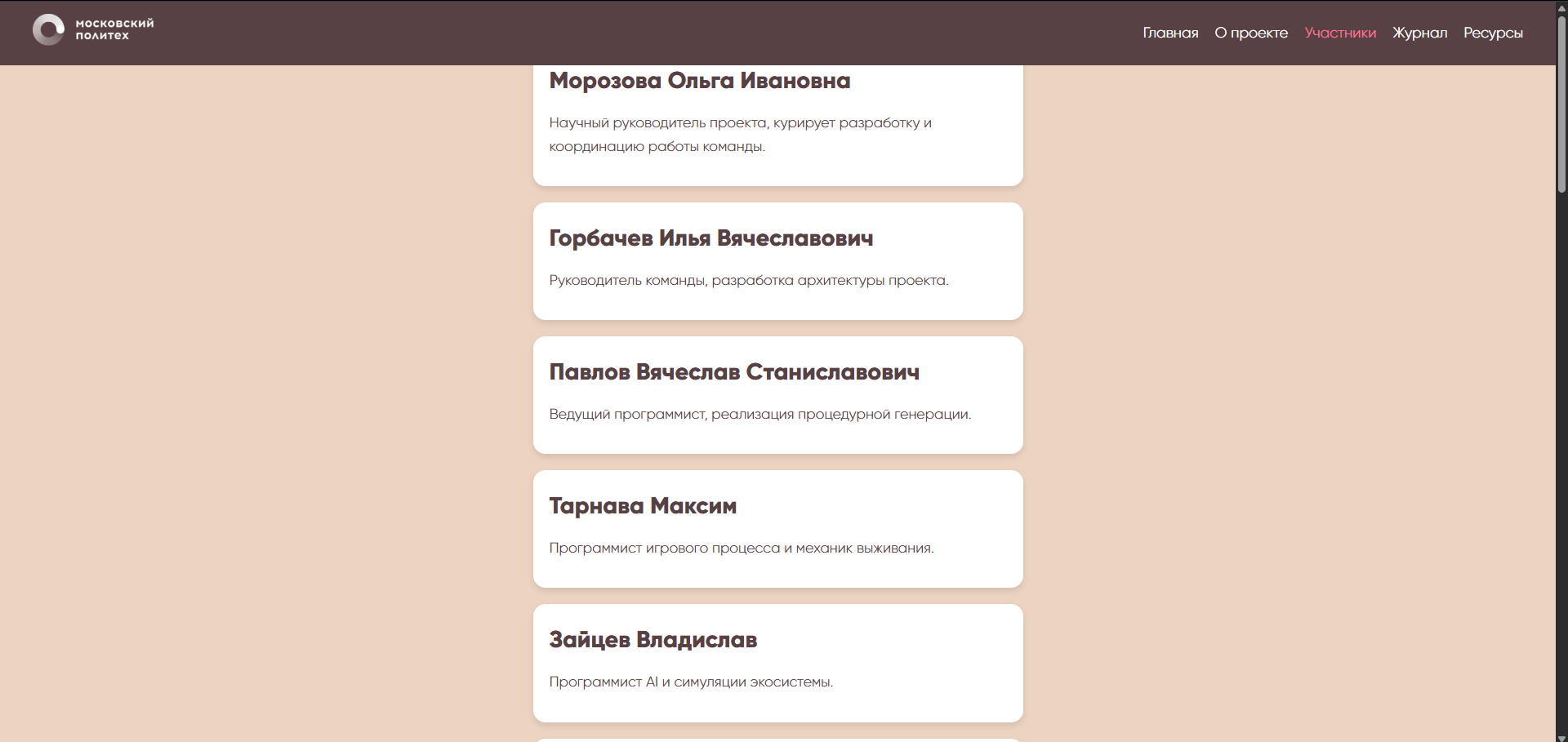


Рисунок 4. Страница «Участники»

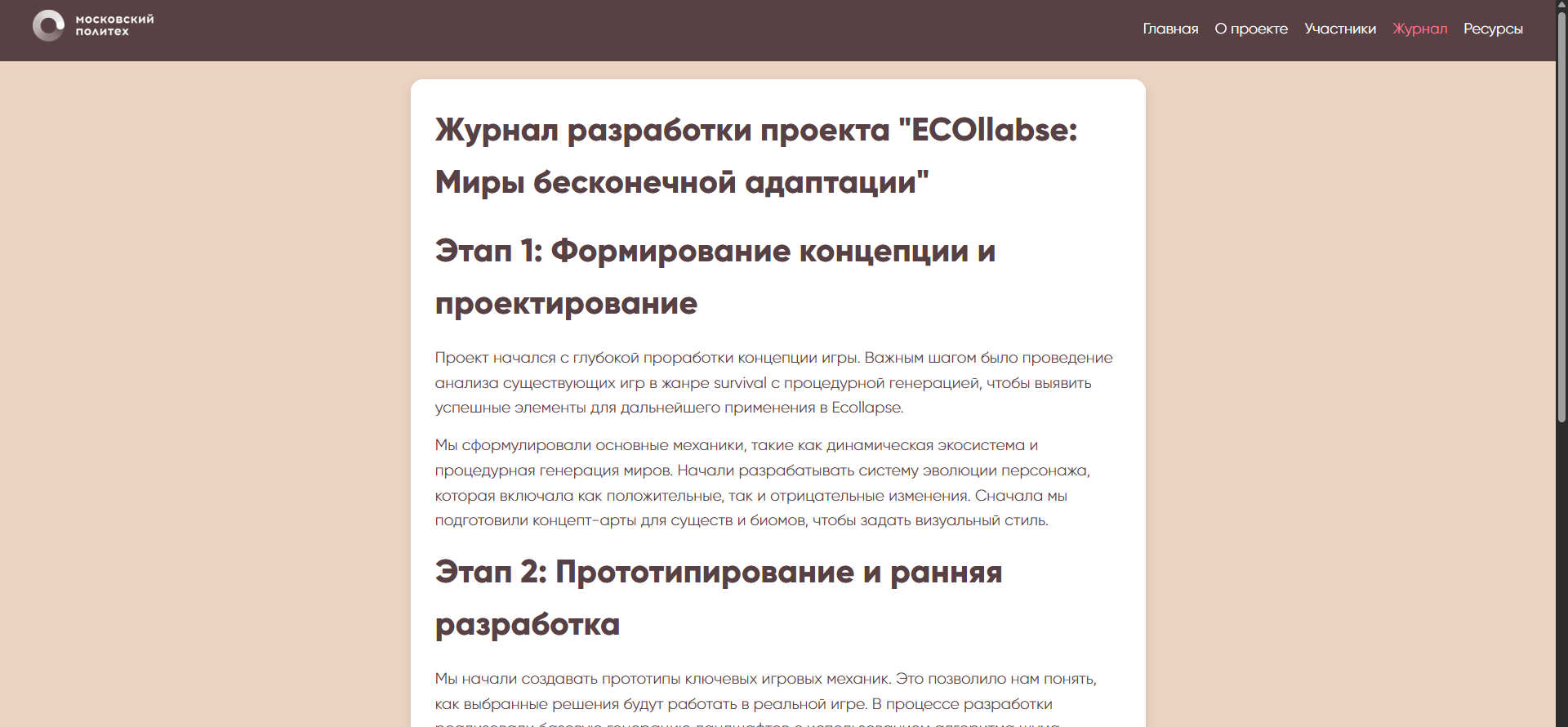


Рисунок 5. Страница «Журнал»

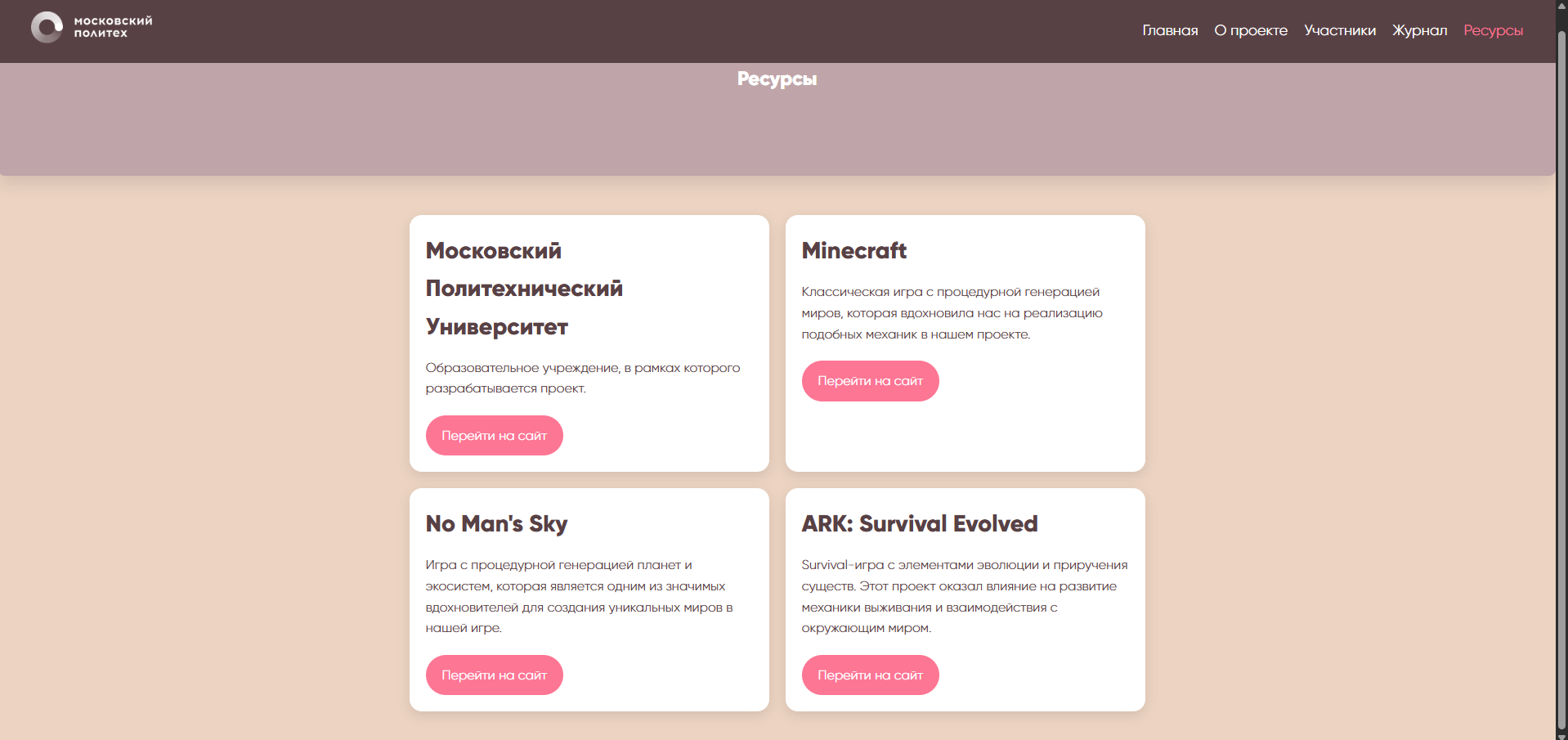


Рисунок 6. Страница «Ресурсы»

#### 4.1.1 Экскурсия в компанию "Первый Бит"

В рамках проектной практики я принял участие экскурсии:

Дата: 27 мая 2025 в 11:00.

Адрес: г. Москва, Воронцовская ул. 35А, корп.1

**Экскурсия в компанию "Первый Бит"**

В компании "Первый Бит" состоялась встреча, на которой я, как и другие участники, познакомился с основными направлениями работы компании. Во время презентации нам рассказали о ключевых технологиях, которые компания использует для автоматизации бизнес-процессов и цифровизации. Особенно интересно было узнать, как ИТ-решения помогают компаниям оптимизировать свою работу и какие современные тенденции в области технологий сейчас доминируют на рынке.

В ходе экскурсии была затронута важная тема трудоустройства в IT-сфере. Спикеры рассказали, какие навыки сейчас наиболее востребованы у специалистов, и какие шаги стоит предпринимать для успешного трудоустройства, даже если у кандидата нет значительного опыта. Было сделано акцент на том, как развивать профессиональные навыки, чтобы стать конкурентоспособным на рынке труда. Особое внимание уделили важности саморазвития и сеть контактов в индустрии.

****

Рисунок 7. Фото на фоне офиса “Первый Бит”

## 5.2 Вариативная часть

### 5.2.1 Описание задачи проекта

В рамках проектной практики я разработал телеграм-бота-помощника для игры «ECOllabse: Миры бесконечной адаптации». Идея бота — помочь новым игрокам лучше понять механику игры, выбрать подходящие мутации, следить за достижениями и адаптироваться к условиям игрового мира.

Бот создаёт удобный интерфейс с меню, через который игрок может выбирать биомы, климат и флору, получать рекомендации по мутациям, добавлять и просматривать свои достижения, а также вести историю своих действий.

Цель проекта - Создать функционального телеграм-бота, который облегчает процесс освоения и игры, предоставляя подсказки и инструменты для планирования эволюции персонажа.

Используемые технологии:

* Язык программирования Python.
* Библиотека pyTelegramBotAPI для взаимодействия с Telegram.
* Хранение данных в JSON-файле.

Задачи:

* Изучить основы работы с Telegram Bot API на Python.
* Реализовать меню с inline-кнопками для выбора параметров мира.
* Создать систему рекомендаций по мутациям и адаптациям.
* Добавить функционал ведения и просмотра достижений.
* Реализовать хранение пользовательских данных и истории действий.
* Обеспечить возможность очистки данных с подтверждением.

### 5.2.2 Пошаговое руководство по созданию

1. Загрузка и сохранение данных пользователей

import json

import os

DATA\_FILE = 'users\_data.json'

# Загрузка данных из файла или создание пустой базы, если файла нет

if os.path.exists(DATA\_FILE):

with open(DATA\_FILE, 'r', encoding='utf-8') as f:

users\_data = json.load(f)

else:

users\_data = {}

def save\_data():

with open(DATA\_FILE, 'w', encoding='utf-8') as f:

json.dump(users\_data, f, ensure\_ascii=False, indent=4)

Этот код отвечает за хранение данных игроков между сессиями. При старте бот загружает информацию из файла JSON, а при изменениях он сохраняет обратно.

2. Получение и инициализация данных пользователя.

def get\_user\_data(user\_id):

uid = str(user\_id)

if uid not in users\_data:

users\_data[uid] = {}

defaults = {

'biome': None,

'climate': None,

'flora': None,

'mutations\_positive': [],

'mutations\_negative': [],

'achievements': [],

'history': []

}

for key, default\_value in defaults.items():

if key not in users\_data[uid]:

users\_data[uid][key] = default\_value

return users\_data[uid]

Функция обеспечивает, что у каждого пользователя есть необходимые поля данных, даже если он только начал пользоваться ботом. Если поле отсутствует, оно создаётся с начальным значением. Это помогает избежать ошибок при работе с данными.

3. Главное меню с интерактивными кнопками

from telebot import types

def main\_menu():

markup = types.InlineKeyboardMarkup(row\_width=2)

markup.add(

types.InlineKeyboardButton("Выбрать биом", callback\_data="choose\_biome"),

types.InlineKeyboardButton("Выбрать климат", callback\_data="choose\_climate"),

types.InlineKeyboardButton("Выбрать флору", callback\_data="choose\_flora"),

types.InlineKeyboardButton("Просмотреть мутации", callback\_data="view\_mutations"),

types.InlineKeyboardButton("Просмотреть достижения", callback\_data="view\_achievements"),

types.InlineKeyboardButton("Добавить достижение", callback\_data="add\_achievement"),

types.InlineKeyboardButton("Просмотреть историю", callback\_data="view\_history"),

types.InlineKeyboardButton("Очистить данные", callback\_data="clear\_data"),

types.InlineKeyboardButton("Помощь", callback\_data="help")

)

return markup

Этот блок создаёт основное меню бота с кнопками для выбора разных функций. Кнопки интерактивные. Нажимая их, пользователь отправляет специальные callback-команды, которые обрабатываются ботом.

4. Обработка выбора биома через inline-кнопки

@bot.callback\_query\_handler(func=lambda call: True)

def callback\_handler(call):

user\_id = call.from\_user.id

user\_data = get\_user\_data(user\_id)

if call.data == "choose\_biome":

kb = types.InlineKeyboardMarkup(row\_width=2)

for biome in BIOMES.keys():

kb.add(types.InlineKeyboardButton(biome.capitalize(), callback\_data=f"set\_biome:{biome}"))

bot.edit\_message\_text("Выбери биом:", call.message.chat.id, call.message.message\_id, reply\_markup=kb)

elif call.data.startswith("set\_biome:"):

biome = call.data.split(":", 1)[1]

user\_data['biome'] = biome

save\_data()

bot.edit\_message\_text(f"Выбран биом: <b>{biome.capitalize()}</b>\n\n{BIOMES[biome]}",

call.message.chat.id, call.message.message\_id, parse\_mode='HTML', reply\_markup=main\_menu())

Здесь реализована логика выбора биома. Сначала бот предлагает выбрать биом из списка кнопок, а потом сохраняет выбор пользователя и выводит информацию с описанием выбранного биома.

5. Добавление достижений и их просмотр

ACHIEVEMENTS = [

"Первый шаг в эволюции",

"Выжил первую ночь",

"Освоил новый биом",

… (и тд. их достаточно много. не стал все вставлять)

elif call.data == "add\_achievement":

kb = types.InlineKeyboardMarkup(row\_width=1)

for i, ach in enumerate(ACHIEVEMENTS):

kb.add(types.InlineKeyboardButton(ach, callback\_data=f"add\_ach:{i}"))

bot.edit\_message\_text("Выбери достижение для добавления:", call.message.chat.id, call.message.message\_id, reply\_markup=kb)

elif call.data.startswith("add\_ach:"):

idx = int(call.data.split(":")[1])

ach\_name = ACHIEVEMENTS[idx]

if ach\_name not in user\_data['achievements']:

user\_data['achievements'].append(ach\_name)

save\_data()

msg = f"Достижение <b>{ach\_name}</b> добавлено!"

else:

msg = "Это достижение уже есть."

bot.edit\_message\_text(msg, call.message.chat.id, call.message.message\_id, parse\_mode='HTML', reply\_markup=main\_menu())

Пользователь может выбрать достижение из списка и добавить его к своим. При попытке добавить уже существующее достижение бот уведомляет об этом.

6. Очистка данных с подтверждением

def clear\_menu():

markup = types.InlineKeyboardMarkup(row\_width=3)

markup.add(

types.InlineKeyboardButton("Удалить последние данные", callback\_data="clear\_last"),

types.InlineKeyboardButton("Удалить все данные", callback\_data="clear\_all\_confirm"),

types.InlineKeyboardButton("Отмена", callback\_data="clear\_cancel")

)

return markup

elif call.data == "clear\_data":

bot.edit\_message\_text("Выберите действие очистки данных:", call.message.chat.id, call.message.message\_id, reply\_markup=clear\_menu())

elif call.data == "clear\_last":

history = user\_data.get('history', [])

if len(history) > 5:

user\_data['history'] = history[:-5]

else:

user\_data['history'] = []

save\_data()

bot.edit\_message\_text("Последние данные удалены.", call.message.chat.id, call.message.message\_id, reply\_markup=main\_menu())

elif call.data == "clear\_all\_confirm":

users\_data[str(user\_id)] = {

'biome': None,

'climate': None,

'flora': None,

'mutations\_positive': [],

'mutations\_negative': [],

'achievements': [],

'history': []

}

save\_data()

bot.edit\_message\_text("Все данные успешно удалены.", call.message.chat.id, call.message.message\_id, reply\_markup=main\_menu())

elif call.data == "clear\_cancel":

bot.edit\_message\_text("Удаление отменено.", call.message.chat.id, call.message.message\_id, reply\_markup=main\_menu())

Реализовано меню очистки с тремя вариантами: удалить последние данные, удалить все данные пользователя или отменить операцию.

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения проектной практики мной был создан телеграм-бот, который помогает игрокам лучше разобраться в игре «ECOllabse: Миры бесконечной адаптации». В процессе работы были выполнены основные задачи.

Освоены базовые возможности работы с Telegram Bot API и Python: обработка команд и callback-запросов, работа с JSON-файлами для сохранения информации, создание интерактивных меню. И ещё приобретён опыт организации пользовательского взаимодействия и управления состоянием данных.

# **CПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Официальная документация Telegram Bot API: https://core.telegram.org/bots/api (дата обращения: 31.05.2025).
2. Документация по библиотеке pyTelegramBotAPI (Telebot): <https://github.com/eternnoir/pyTelegramBotAPI> (дата обращения: 21.05.2025).
3. Официальный сайт Python: <https://www.python.org/doc/> (дата обращения: 31.05.2025).
4. Руководство по JSON в Python: <https://docs.python.org/3/library/json.html> (дата обращения: 21.05.2025).
5. Официальное руководство по Git: <https://git-scm.com/doc> (дата обращения: 21.05.2025).
6. MDN Web Docs - HTML и JavaScript (для понимания inline-кнопок): <https://developer.mozilla.org/ru/docs/Web/API/Element/click_event> (дата обращения: 21.05.2025).
7. Статья по созданию Telegram-ботов на Python: https://realpython.com/how-to-make-a-discord-bot-python/ (дата обращения: 31.05.2025).
8. Руководство по Markdown: <https://www.markdownguide.org/basic-syntax/> (дата обращения: 19.05.2025).