Федеральное государственное автономное образовательное учреждение высшего образования
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий
Кафедра «Информатика и информационные технологии»

Направление подготовки/ специальность: АСОИУ_____

ОТЧЕТ

по проектной практике

Студент: Дунаева Ева_____ 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

Общая информация о проекте:

Проект «Лаболатория успеха»

Цели и задачи проекта

Разработка и реализация различных методов применения геймификации в формате различных настольных и компьютерных игр, позволяющих игроку применить и повысить свои управленческие, административные и профессиональные навыки и увеличить его стремление к дальнейшему развитию.

Разработать варианты применения геймификации в сфере дополнительного образования кадров и создать прототип.

Задачи:

1. Проанализировать рынок, текущие предложения, аудиторию;
2. Создать игровую концепцию, направленную на мотивацию развития
1. профессиональных навыков и личностных качеств, обучающую основам
2. менеджмента и планирования;
3. Разработать прототип игры;
4. Презентовать полученные данные и вариант реализации.
5. Создать сайт с информацией о проекте
6.

Проект **«Лаборатория успеха»** направлен на разработку геймифицированных решений, ориентированных на развитие профессиональных компетенций. Его главная цель — **создание линейки онлайн и настольных игр**, предназначенных для моделирования производственных и проектных ситуаций, повышения мотивации к обучению и командному взаимодействию.

Основные задачи проекта

- Разработка **продуктовой линейки** игр с гибкой структурой, позволяющей масштабировать продукт и адаптировать его под разные аудитории.
- Повышение **мотивации к развитию профессиональных навыков** через игровые механики и формат состязаний.
- Формирование навыков **решения прикладных задач** в управлении, экономике, коммуникации и производственной сфере.
- Реализация **игровых форматов** в как цифровом, так и аналоговом исполнении.

Формы применения игровых решений

Разрабатываемые игры могут использоваться для следующих типов активностей:

1. Командные состязания с предметным содержанием
2. Цифровая геймификация образовательных и производственных процессов
3. Корпоративные обучающие игры
4. Сезонные турниры профессионального мастерства
5. Игровой ассесмент (оценка навыков через игру)
6. Наставничество и обучение отдельным компетенциям

Преимущества подхода

- **Модульная структура** игровых решений — адаптация под конкретные профессиональные задачи
- **Единая игровая платформа** и движок, объединяющие онлайн и настольные версии
- **Уникальный визуальный стиль** и интерфейс
- **Возможность как индивидуальной, так и командной игры**
- **Гибкая архитектура** для масштабирования и обновления

Этапы реализации проекта

1. Разработка **прототипа фрагмента профессиональной игры** для демонстрации и привлечения заказчиков
2. Выпуск серии **настольных игр** для командообразования и профессионального развития
3. Создание **полноценной онлайн-игры** для предметных областей (например: полиграфия, упаковка, управление персоналом)
4. Разработка серии **производственных симуляций** для индустриального сектора
5. Интеграция игровых решений в **систему образования** и подготовку кадров

Направления работ

Проект реализуется по двум ключевым направлениям:

1. Модульная онлайн-игра

- Исследование аудитории, формирование концепции и требований
- Разработка гейм-дизайна, интерфейсов, библиотеки объектов
- Прототипирование игровых сценариев
- Внедрение игровых механик: симуляция, обучение, рейтинг, достижения
- Тестирование, запуск и поддержка

Жанры и особенности:
Симулятор / стратегия / ролевая обучающая игра
– Реалистичная симуляция профессиональных процессов
– Система прогресса, достижений, соревновательные элементы
– Интерактивные сценарии с реальными задачами

2. Серия настольных игр

- Анализ потребностей целевой аудитории
- Создание стратегии, сценариев, механик и компонентов
- Настольные игры с тренером и без него

Жанры:
Образовательная стратегия, симулятор профессиональных задач, производственный квест, кооперативная ролевая игра

Особенности:
– Моделирование профессиональных ситуаций
– Принятие решений и их влияние на общий результат
– Обучение через взаимодействие

Вариативная часть задания
Цели и задачи проекта

Разработка и реализация базовой модели **блокчейн-системы** на языке программирования **Python** с использованием библиотеки Flask для создания API. Цель проекта — продемонстрировать ключевые принципы работы блокчейна, включая создание блоков, добавление транзакций, майнинг и синхронизацию узлов сети (консенсус).

Задачи проекта

1. **Изучить основы блокчейна**
   Ознакомиться с теоретическими принципами работы технологии блокчейн: хеш-функции, Proof of Work, структура блока, транзакции, консенсус и распределённая сеть.
2. **Реализовать класс Blockchain на Python**
   Создать класс, содержащий:
   - цепочку блоков;
   - методы для добавления новых транзакций;
   - алгоритм майнинга (доказательство работы);
   - метод генерации новых блоков;

- валидацию цепочки.

3. **Настроить REST API с помощью Flask**
   Разработать API-интерфейс для взаимодействия с блокчейном:
   - добавление транзакций
   - запуск майнинга
   - просмотр всей цепочки
   - регистрация и синхронизация узлов
4. **Организовать взаимодействие между несколькими узлами**
   Проверить корректность работы алгоритма консенсуса при наличии двух и более узлов. Обеспечить возможность синхронизации данных между нодами.
5. **Протестировать работу проекта с помощью Postman**
   Проверить корректную работу всех эндпоинтов API, включая отправку транзакций и консенсус между нодами.

Хронология выполнения проекта

1. Постановка цели и задач

В начале проекта была выбрана тема — разработка базовой блокчейн-системы. Определены основные цели: понять архитектуру блокчейна, реализовать работу узлов, консенсус и взаимодействие через API.

2. Изучение предметной области

Проведено исследование технологии блокчейн. Изучены следующие ключевые понятия:

Структура блока

Хеш-функции

Принцип Proof of Work

Механизмы децентрализации и консенсуса

Работа с Flask и REST API

В качестве источников использованы статьи и туториалы, в том числе Learn Blockchains by Building One.

3. Реализация проекта

Создана базовая структура блокчейна:

Класс Blockchain для хранения цепочки блоков

Метод для добавления транзакций

Реализация механизма майнинга (Proof of Work)

Создание REST API с помощью Flask (endpoints: /mine, /transactions/new, /chain)

Добавлены функции децентрализации:

Регистрация узлов сети

Разрешение конфликтов (реализация алгоритма консенсуса)

4. Тестирование

Проект протестирован с использованием Postman. Запускались два и более узлов локально, проверялась синхронизация цепочки между ними..

Заключение

Проект позволил участникам познакомиться с основами блокчейна, реализовать и протестировать собственную сеть. Получен ценный опыт в области Python, сетевого взаимодействия и создания технической документации. В дальнейшем проект может быть расширен за счет хранения данных в базе, авторизации и визуального интерфейса.

Листинг программы:

```
import hashlib
import json
from time import time
from urllib.parse import urlparse
from uuid import uuid4blo

import requests
from flask import Flask, jsonify, request


class Blockchain:
    def __init__(self):
        self.current_transactions = []
        self.chain = []
        self.nodes = set()

        self.new_block(previous_hash='1', proof=100)

    def register_node(self, address):

        parsed_url = urlparse(address)
        if parsed_url.netloc:
            self.nodes.add(parsed_url.netloc)
        elif parsed_url.path:
            self.nodes.add(parsed_url.path)
        else:
            raise ValueError('Неверный URL')

    def valid_chain(self, chain):

        last_block = chain[0]
        current_index = 1

        while current_index < len(chain):
```

```
            block = chain[current_index]
            print(f'{last_block}')
            print(f'{block}')
            print("\n-----------\n")

            last_block_hash = self.hash(last_block)
            if block['previous_hash'] != last_block_hash:
                return False


            if not self.valid_proof(last_block['proof'], block['proof'],
last_block_hash):
                return False

            last_block = block
            current_index += 1

        return True

    def resolve_conflicts(self):

        neighbours = self.nodes
        new_chain = None


        max_length = len(self.chain)

        for node in neighbours:
            response = requests.get(f'http://{node}/chain')

            if response.status_code == 200:
                length = response.json()['length']
                chain = response.json()['chain']

                if length > max_length and self.valid_chain(chain):
                    max_length = length
                    new_chain = chain


        if new_chain:
            self.chain = new_chain
            return True

        return False

    def new_block(self, proof, previous_hash):

        block = {
            'index': len(self.chain) + 1,
            'timestamp': time(),
            'transactions': self.current_transactions,
            'proof': proof,
            'previous_hash': previous_hash or self.hash(self.chain[-1]),
        }

        self.current_transactions = []

        self.chain.append(block)
        return block

    def new_transaction(self, sender, recipient, amount):

        self.current_transactions.append({
```

```
            'sender': sender,
            'recipient': recipient,
            'amount': amount,
        })

        return self.last_block['index'] + 1

    @property
    def last_block(self):
        return self.chain[-1]

    @staticmethod
    def hash(block):

        block_string = json.dumps(block, sort_keys=True).encode()
        return hashlib.sha256(block_string).hexdigest()

    def proof_of_work(self, last_block):

        last_proof = last_block['proof']
        last_hash = self.hash(last_block)

        proof = 0
        while self.valid_proof(last_proof, proof, last_hash) is False:
            proof += 1

        return proof

    @staticmethod
    def valid_proof(last_proof, proof, last_hash):


        guess = f'{last_proof}{proof}{last_hash}'.encode()
        guess_hash = hashlib.sha256(guess).hexdigest()
        return guess_hash[:4] == "0000"


app = Flask(__name__)

node_identifier = str(uuid4()).replace('-', '')


blockchain = Blockchain()


@app.route('/mine', methods=['GET'])
def mine():

    last_block = blockchain.last_block
    proof = blockchain.proof_of_work(last_block)


    blockchain.new_transaction(
        sender="0",
        recipient=node_identifier,
        amount=1,
    )


    previous_hash = blockchain.hash(last_block)
    block = blockchain.new_block(proof, previous_hash)

    response = {
```

```
        'message': "Создан новый блок",
        'index': block['index'],
        'transactions': block['transactions'],
        'proof': block['proof'],
        'previous_hash': block['previous_hash'],
    }
    return jsonify(response), 200


@app.route('/transactions/new', methods=['POST'])
def new_transaction():
    values = request.get_json()


    required = ['sender', 'recipient', 'amount']
    if not all(k in values for k in required):
        return 'Не хватает значений', 400


    index = blockchain.new_transaction(values['sender'], values['recipient'],
values['amount'])

    response = {'message': f'Транзакция будет добавлена в блок {index}'}
    return jsonify(response), 201


@app.route('/chain', methods=['GET'])
def full_chain():
    response = {
        'chain': blockchain.chain,
        'length': len(blockchain.chain),
    }
    return jsonify(response), 200


@app.route('/nodes/register', methods=['POST'])
def register_nodes():
    values = request.get_json()

    nodes = values.get('nodes')
    if nodes is None:
        return "Ошибка: Пожалуйста, предоставьте действительный список нодов", 400

    for node in nodes:
        blockchain.register_node(node)

    response = {
        'message': 'Новые ноды были добавлены',
        'total_nodes': list(blockchain.nodes),
    }
    return jsonify(response), 201


@app.route('/nodes/resolve', methods=['GET'])
def consensus():
    replaced = blockchain.resolve_conflicts()

    if replaced:
        response = {
            'message': 'Our chain was replaced',
            'new_chain': blockchain.chain
        }
    else:
```

```
        response = {
            'message': 'Our chain is authoritative',
            'chain': blockchain.chain
        }

    return jsonify(response), 200


if __name__ == '__main__':
    from argparse import ArgumentParser

    parser = ArgumentParser()
    parser.add_argument('-p', '--port', default=5001, type=int, help='port to
listen on')
    args = parser.parse_args()
    port = args.port

    app.run(host='0.0.0.0', port=port)
```

Общая характеристика деятельности организации *(заказчика проекта)*

ITE Group

Организационная структура:
ITE Group — международная компания, специализирующаяся на организации отраслевых выставок, конференций и деловых мероприятий. С момента основания в 1991 году компания выстроила эффективную структуру, включающую:

- **Отделы по направлениям отраслей**, ответственные за планирование и проведение мероприятий в различных секторах экономики.
- **Маркетинговый отдел**, занимающийся продвижением мероприятий и привлечением участников.
- **Технический отдел**, обеспечивающий цифровую поддержку и инновационные решения для проведения мероприятий.
- **Отдел клиентского сервиса**, обеспечивающий высокое качество обслуживания участников и посетителей.

Компания имеет представительства в различных регионах, что позволяет эффективно координировать мероприятия на международном уровне.

Описание деятельности:
ITE Group является одним из ведущих организаторов международных торговых выставок и конференций, специализируясь на мероприятиях в развивающихся и растущих рынках. Основная деятельность компании направлена на:

- Организацию более 240 выставок и конференций ежегодно в 14 странах, охватывая ключевые отрасли экономики.
- Создание платформ для обмена опытом, презентации инновационных решений и установления деловых контактов.
- Поддержку развития бизнеса и профессионального роста участников через насыщенные деловые программы и мероприятия.

Цифровые решения:
В рамках цифровой трансформации ITE Group разработала платформу **ITE Connect**, которая служит отраслевым сообществом для делового общения. Платформа предоставляет:

- Возможность обмена новостями, экспертизой и аналитикой индустрии.
- Проведение профильных онлайн-мероприятий, таких как вебинары и воркшопы.
- Инструменты для установления и поддержания деловых контактов в онлайн-среде.

Миссия ITE Group — создавать уникальные события и цифровые решения, способствующие успеху бизнеса и развитию отраслей экономики. Компания стремится предоставлять эффективные и инновационные инструменты для поиска и установления деловых контактов, развития бизнеса и профессионального роста.

Описание задания по проектной практике

1) Проведен анализ возможностей команды, исследование рынка, аудитории и возможных вариантов реализации проекта.
Распределена структура работы проекта, принято решение по текущей разработке.
Сформулированы основные механики игр, их концепция.
Проделана работа над визуальным наполнением игр.
Сделаны первые прототипы настольной игры, начаты наработки видеоигры.
Проведены первые тестирования настольной игры, внесены изменения.
В ходе работы над проектом была создана настольная экономическая игра основанная на механике расстановки рабочих и зданий, направленная на стратегическое мышление, управление ресурсами, планирование развития и взаимодействие с другими игроками.
Основной целью игры является построение
успешной компании, конкурируя с другими игроками.
Конкуренция за локации и ресурсы, бонусы за сотрудничество и необходимость взвешивания рисков и наград, заставляют игроков принимать сложные решения и стратегически планировать свои действия.
Кроме этого, была начата разработка видеоигры о развитии промышленных цепочек, планировании и менеджменте собственного бизнеса с возможностью многопользовательской игры и конкуренции. Получены первые наработки.

2) Разработан прототип blockchain, включающий:

- базовый функционал создания цепочки блоков;
- обработку транзакций;
- алгоритм майнинга и генерации новых блоков;
- проверку целостности цепи и валидности доказательства работы;
- систему регистрации и синхронизации узлов сети с использованием REST API.

Проведено локальное тестирование нескольких узлов с помощью Postman, подтверждена корректная работа механизмов консенсуса. Реализована возможность запуска нескольких экземпляров блокчейн-сервера на разных портах для моделирования распределённой сети.

Проект направлен на образовательные цели, помогает освоить фундаментальные концепции децентрализованных сетей, криптографической безопасности и взаимодействия API. В рамках практики также обсуждались перспективы расширения: внедрение смарт-контрактов, визуализация цепи блоков, создание веб-интерфейса.

ЗАКЛЮЧЕНИЕ

За время работы в семестре были расставлены задачи, цели проекта, построено представление желаемого результата. В ходе работы были получены прототипы продуктовых результатов, проведены тестирования, внесены изменения. Участники показали хороший уровень подготовки и вовлеченность в проект. В дальнейшем планируется продолжение развития настольной игры и разработка видеоигры.

Также в течение семестра была сформулирована цель проекта — создание базового прототипа блокчейн-системы с возможностью регистрации транзакций, майнинга блоков и синхронизации между узлами сети. Определены задачи, необходимые для достижения цели, и выстроено представление о структуре и функциональности системы.

В процессе работы были реализованы основные компоненты блокчейна: создание цепочки блоков, реализация алгоритма доказательства выполнения работы (Proof of Work), регистрация узлов и механизм консенсуса для синхронизации цепочки. Проведено локальное тестирование работы системы с использованием нескольких узлов, в том числе через Postman и браузерные интерфейсы.

Ссылка на Гитхаб: https://github.com/dunaevaEva/projectWork_DE

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. **Введение в CSS-верстку**
   https://developer.mozilla.org/ru/docs/Learn_web_development/Core/CSS_layout/Introduction
2. **DevTools для «чайников» (обзор возможностей браузерной отладки)**
   https://habr.com/ru/articles/548898/
3. **Элементы HTML**
   https://developer.mozilla.org/ru/docs/Web/HTML/Element
4. **Основы HTML: создание первого сайта**
   https://developer.mozilla.org/ru/docs/Learn_web_development/Getting_started/Your_first_website/Creating_the_content
5. **Основы CSS**
   https://developer.mozilla.org/ru/docs/Web/CSS
6. **Doka.guide — справочник по веб-разработке**
   https://doka.guide/
7. **Официальная документация по Git**
   https://git-scm.com/book/ru/v2
8. **Git для начинающих: объяснение на схемах (Skillbox)**
   https://skillbox.ru/media/code/chto_takoe_git_obyasnyaem_na_skhemakh/
9. **Введение в Git (курс на Hexlet)**
   https://ru.hexlet.io/courses/intro_to_git
10. **Уроки по Markdown (Hexlet)**
    https://ru.hexlet.io/lesson_filters/markdown
11. **Learn Blockchains by Building One (статья на Medium)**
    https://hackernoon.com/learn-blockchains-by-building-one-117428612f46
12. **Build Your Own Blockchain**
    http://ecomunsing.com/build-your-own-blockchain