

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
ИМЕНИ В.Ф. УТКИНА»

Кафедра «Вычислительной и прикладной математики»

**ОТЧЕТ**

о прохождении технологической (производственно-технологической)  
практики

студента 3 курса, группы 143

Соколова Егора Александровича  
(Ф.И.О. студента)

Вид практики производственная Семестр 6  
(учебная, производственная, преддипломная, технологическая)

Тип практики стационарная  
(стационарная, выездная)

Сроки прохождения практики:

с «21» июня 2024 г. по «18» июля 2024 г.

Продолжительность практики 4  
(недель)

Трудоемкость практики 216/6  
(часов / зач.ед.)

Место прохождения практики ООО «РНТ»

Занимаемая должность студент–практикант

Руководитель практики от предприятия (организации, учреждения)

Силкин Григорий Дмитриевич

Руководитель практики от университета

Доцент, к.т.н., Макаров Николай Петрович

(должность, ФИО)

Оценка \_\_\_\_\_

(Подпись)

## ЗАДАНИЕ НА ПРОИЗВОДСТВЕННУЮ ПРАКТИКУ

студенту 3 курса, группы 143

Соколову Егору Александровичу  
(Ф.И.О.)

В ходе выполнения производственной практики студенту требуется:

- 1 Ознакомиться с работой организации «RNT Group»
- 2 Ознакомиться с правилами организации и с основными требованиями техники безопасности организации.
- 3 Изучение введения в жизненный цикл информационных систем и методологии разработки
- 4 Вводная лекция и практическое задание по бизнес и системному анализу – работа с пользовательскими историями
- 5 Лекция по клиент-серверной архитектуре и как устроен бекенд приложения
- 6 Практическая работа по frontend-части (HTML+CSS+JS)
- 7 Тестирование и автотестирование, теория и практическая работа с поcode инструментами автотестирования
- 8 Настройка BI дашборда и лекция по работе с данными
- 9 Изучение введения в Devops
- 10 Подготовка, оформление отчёта о прохождении практики.

Руководитель практики  
от предприятия: Силкин Григорий Дмитриевич руководитель учебного  
центра /



(подпись)

Согласовано  
руководитель практики от университета

(подпись)

## СОДЕРЖАНИЕ

ЗАДАНИЕ НА ПРОИЗВОДСТВЕННУЮ ПРАКТИКУ .....	2
1 Вводная лекция.....	4
2 Бизнес и системный анализ.....	5
2.1 Техническое задание.....	5
2.2 Реализованные истории пользователя .....	5
3 Проектирование макета интерфейса пользователя .....	8
3.1 Техническое задание.....	8
3.2 Обновленные прототипы интерфейса приложения.....	9
4 Реализация интерфейса пользователя на HTML.....	13
4.1 Техническое задание.....	13
4.2 Реализованная WEB–страница .....	14
5 Добавление к WEB–странице таблиц стилей CSS .....	16
5.1 Техническое задание.....	16
5.2 Результат выполнения задания .....	16
6 Реализация frontend–логики на языке JavaScript .....	20
6.1 Техническое задание.....	20
6.2 Реализация задания .....	21
7 Тестирование серверной части WEB-приложения.....	25
7.1 Техническое задание.....	25
7.2 Результаты тестирования .....	26
8 Анализ разработанного приложения при помощи практик Business Intelligence .....	28
8.2 Техническое задание.....	28
8.2 Результат выполнения задания .....	30
9 Дополнительные лекции.....	32
ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ ПРАКТИКИ .....	33
БИБЛИОГРАФИЧЕСКИЙ СПИСОК .....	37
ОТЗЫВ .....	38

## **1 Вводная лекция**

Во время вводной лекции был проведен инструктаж по технике безопасности, уточнены детали проведения практики (процесс сдачи домашних заданий, взаимодействий с кураторами от предприятия и т.п.). Также были рассмотрены основные методологии разработки ПО:

### **I) Водопадная модель разработки ПО**

Этапы:

- 1) Анализ требований;
- 2) Дизайн системы;
- 3) Реализация (кодирование);
- 4) Тестирование;
- 5) Внедрение;
- 5) Поддержка.

Преимущества данной модели – четкая структура и конкретные сроки реализации ПО, главный недостаток – низкая гибкость (сложность выявления и исправления проблем во время разработки).

### **II) Итеративная модель**

Этапы:

- 1) Планирование и анализ;
- 2) Дизайн;
- 3) Реализация;
- 4) Тестирование;
- 5) Оценка и обратная связь.

Главное преимущество данной модели – раннее выявление и исправление ошибок, недостатки – сложность управления итерациями и необходимость постоянного взаимодействия с заказчиком.

### **III) Гибкие методологии (Agile и фреймворк Scrum)**

Agile — гибкий подход к разработке ПО, Scrum — это фреймворк в Agile, разделяющий процесс на спринты (чаще всего продолжаются 2 – 4 недели).

Цикл Scrum:

- 1) Спринт-планирование;
- 2) Ежедневные встречи;
- 3) Разработка и тестирование;
- 4) Спринт-ревью;
- 5) Ретроспектива.

Преимущества – высокая гибкость и адаптивность, частые релизы и ранняя обратная связь, недостатки – необходимость постоянной вовлеченности заказчика и сложности с масштабированием на слишком большие проекты.

## 2 Бизнес и системный анализ

### 2.1 Техническое задание

Требуется разработать WEB–приложение «Полицейские и воры». Серверная часть приложения уже реализована. Необходимо подготовить описание сценария (историй пользователя) для новых функций:

- 1) Создание новой игры;
- 2) Управление игрой;
- 3) Управление сессией игры (подключение/отключение);
- 4) Действия в игре;
- 5) Отображение статистики по игре.

### 2.2 Реализованные истории пользователя

**1. История пользователя:** как пользователь, я хочу создать новую игровую комнату, чтобы поиграть с друзьями или другими пользователями

**Функция системы:** создание новой игры

**Критерии приёмки:**

- Пользователь может создать новую комнату
- Система должна проверить, что:
  - Пользователь авторизован в системе
  - Название создаваемой комнаты уникально
- Пользователь выбирает карту для создаваемой комнаты
  - Если желаемая карта уже существует, то пользователь выбирает ее из каталога доступных файлов карт
  - Если желаемой карты не существует, то пользователь создает ее в конструкторе
- Пользователь настраивает параметры комнаты:
  - Числовое значение таймера в секундах для автоматической смены команд местами (по умолчанию – 60 секунд)
  - Количество жизней у роли «Жулик» (по умолчанию – 4)
  - Скорость жуликов (по умолчанию – 1)
  - Скорость полицейских (по умолчанию на 10% больше, чем у жуликов)
  - Система проверяет, что параметры заданы корректно
    - Если один или несколько параметров заданы некорректно, то Система требует от пользователя исправить его/их
    - Если все параметры заданы верно, то комната создана и игра готова к запуску.

**2. История пользователя:** как пользователь, я хочу управлять ходом игры, чтобы иметь возможность присоединиться к ней, ставить ее на паузу, отменять, покидать.

**Функция системы:** управление игрой

**Критерии приёмки:**

- Создатель игровой комнаты может запустить игру в случае, если:
  - В каждой команде набралось как минимум по одному игроку
- При запуске игры в течение пяти секунд отображается обратный отсчет
- Пользователь может
  - Присоединиться к игре, если:
    - Присутствуют свободные места хотя бы в одной из команд и:
      - Игра ещё не началась

- Игра запускается, идет обратный отсчет
    - Покинуть игру
- Создатель комнаты может:
  - Поставить игру на паузу
  - Отменить запущенную игру. В этом случае игра останавливается, команды распускаются, игровая комната удаляется;
  - Покинуть игру, тогда:
    - Если в комнате остался хотя бы один пользователь, то роль создателя команды назначается пользователю, первым после создателя вошедшим в игровую комнату
    - Если пользователей в комнате нет, то она удаляется.

**3. История пользователя:** как пользователь, я хочу управлять сессией игры, чтобы иметь возможность подключаться к игре и выходить из нее

**Функция системы:** управление сессией игры (подключение/отключение)

**Критерии приёмки:**

- Пользователь может подключиться к игре, если:
  - Игра еще не началась
  - Игровая комната не заполнена (хотя бы в одной команде есть свободные места)
- Если требования для присоединения пользователя к игре не выполнены, то для него выводится сообщение о невозможности присоединения с указанием причины
- Если пользователь присоединяется к игре, то:
  - Пользователь выбирает команду полицейских или жуликов.
- Если максимальное количество человек в определенной роли занято, то система:
  - Не позволит пользователю присоединиться к данной роли и выведет соответствующее сообщение пользователю
  - Предложит пользователю еще раз выбрать команду для присоединения
- Пользователь может выйти из игры, тогда:
  - Игровой персонаж вышедшего пользователь останавливается и не принимает участия в игре, другие игроки никак с ним не взаимодействуют
  - Если в комнате осталось как минимум по одному жулику и полицейскому, то игра продолжается в обычном режиме
  - Если в комнате остался один игрок, либо если остались пользователи одного типа, то игра продолжается до тех пор, пока не будут собраны все монеты
  - Если все пользователи комнаты вышли из игры, то игра отменяется.

**4. История пользователя:** как пользователь, я хочу иметь возможность управлять персонажем для повышения динамики игрового процесса

**Функция системы:** действия в игре

**Критерии приёмки:**

- Пользователь может управлять направлением движения персонажа по игровому полю при помощи соответствующих стрелок на клавиатуре
- Игровое поле состоит из клеток, одни из которых свободны для перемещения, другие представляют собой непроходимые стены

- Игровые персонажи бывают двух видов: жулики и полицейские
- В случае, если игровой персонаж натывается на стену, то он останавливается. Пользователь может снова запустить перемещение пользователя стрелками в сторону свободных клеток
- Если игровой персонаж перемещается через проходимую клетку, то он продолжает движение в прежнем направлении. Проходимая клетка может:
  - Быть пустой
  - Содержать монету. Если персонаж:
    - Жулик, то монета с карты исчезает, а на баланс его команды прибавляется +1 монета
    - Полицейский, то для него данная клетка по поведению является пустой, монета не забирается
  - Содержать бонусную жизнь. Если персонаж:
    - Жулик, то бонусная жизнь с карты исчезает, а на баланс его команды прибавляется +1 жизнь
    - Полицейский, то для него данная клетка по поведению является пустой, жизнь не добавляется
  - Менять роли пользователей местами (все жулики становятся полицейскими, а все полицейские – жуликами)
- В начале игры пользователи появляются на точке возрождения соответственно для жуликов и полицейских
- Если полицейский сталкивается с жуликом, то у команды жуликов отнимается одна жизнь, а пойманный жулик отправляется на точку возрождения жуликов
- При срабатывании таймера роли пользователей в команде меняются местами
- Игра завершается, если:
  - Жулики собрали все монеты
  - Полицейские поймали всех жуликов, и у команды жуликов закончились жизни
- Выигрывает команда, у которой в конце игры на счету оказалось больше собранных монет.

**5. История пользователя:** как пользователь, я хочу видеть статистику для команд для наблюдения за ходом игры и просмотром промежуточных результатов

**Функция системы:** отображение статистики по игре

**Критерии приёмки:**

- Пользователь может видеть:
  - Название текущей комнаты, в которой он находится
  - Название текущей команды, за которую он играет
  - Состав команд жуликов и полицейских (логины пользователей)
  - Текущее значение таймера для смены ролей в команде
  - Для каждой команды:
    - Количество очков (собранных монет)
    - Количество оставшихся жизней
- При окончании игры пользователь видит список пользователей команды победителей.

### **3 Проектирование макета интерфейса пользователя**

#### **3.1 Техническое задание**

Обновить визуальный дизайн игры, улучшить стилистику элементов, учитывая современные тренды.

##### **Описание игрового процесса**

Команды собирают призы в одном и том же лабиринте. Жулики имеют цель собрать призы, в то время как полицейские пытаются их остановить, преследуя и "арестовывая". Скорость жуликов должна быть немного выше скорости полицейских. Команды меняются ролями через заданные интервалы времени, продолжая игру в том же лабиринте.

##### **Игровой интерфейс и статистика**

Во время игры и после ее завершения на экране пользователя отображается статистика по командам и каждому игроку.

##### **Условия окончания игры**

- а) Все призы в лабиринте собраны. В этом случае побеждает та команда, которая собрала больше призов;
- б) Все жулики арестованы. В этом случае побеждает команда полицейских. Количество собранных точек в данном случае не играет роли;
- в) Игра принудительно завершена.

##### **Требования к редизайну**

- 1) Обновить визуальный дизайн лабиринта, улучшить стилистику элементов, учитывая тренды;
- 2) Оптимизировать интерфейс для интуитивно понятного управления и отображения статистики по очкам, игрового таймера;
- 3) Показать механику смены ролей, чтобы процесс был плавным и не вызывал путаницы у игроков;
- 4) Разрешение экрана 1920x1080 pxls.

##### **Инструменты:**

Интерфейс проектировать в Figma [1], все артефакты выгрузить отдельным файлом или просто передать ссылку на макеты.

##### **Критерии приемки:**

Кол-во экранов: должно быть 3 артефакта:

- 1) Лабиринт со статистикой и таймером;
- 2) Отрисованные элементы (клетка лабиринта, иконки жулика и полицейского, накопленные жизни, смена ролей, призы для жуликов);
- 3) Уведомление на лабиринте что игроки поменялись ролями.

Интерфейс, требующий обновления, представлен на рисунке 1.



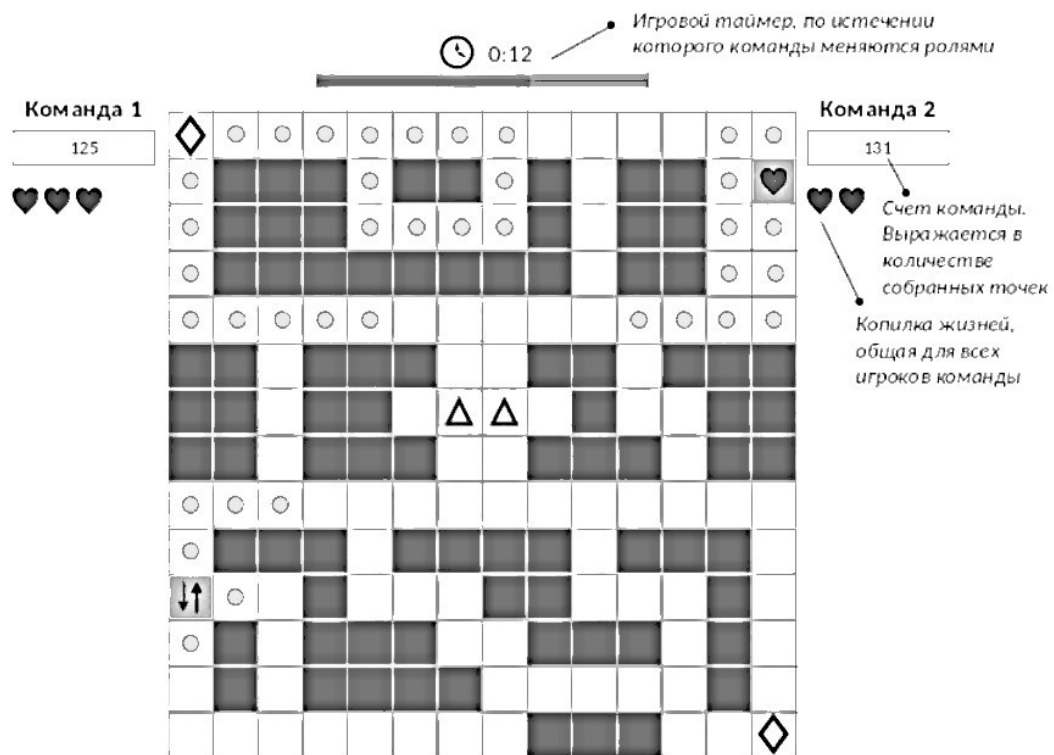


Рисунок 1 – Интерфейс, требующий обновления

### 3.2 Обновленные прототипы интерфейса приложения

Прототипы обновленных интерфейсов для лабиринта, его составных элементов и сигнала о смене ролей представлен на рисунках 2 – 4.

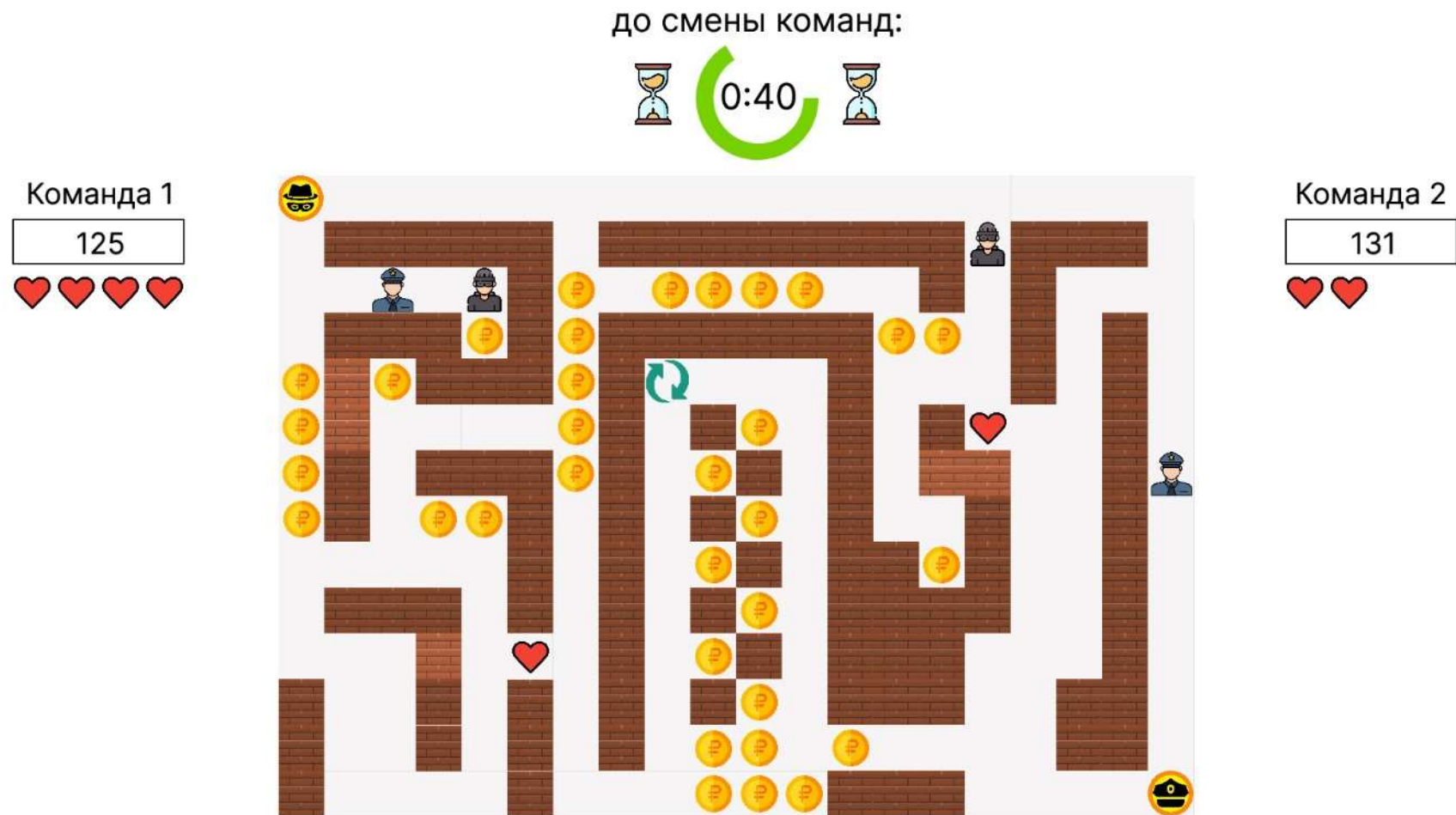


Рисунок 2 – Обновленный прототип пользовательского интерфейса.



Пустая клетка



Стена



Полицейский



Жулик



Точка смены команд



Монета



Жизнь



Точка появления полиции



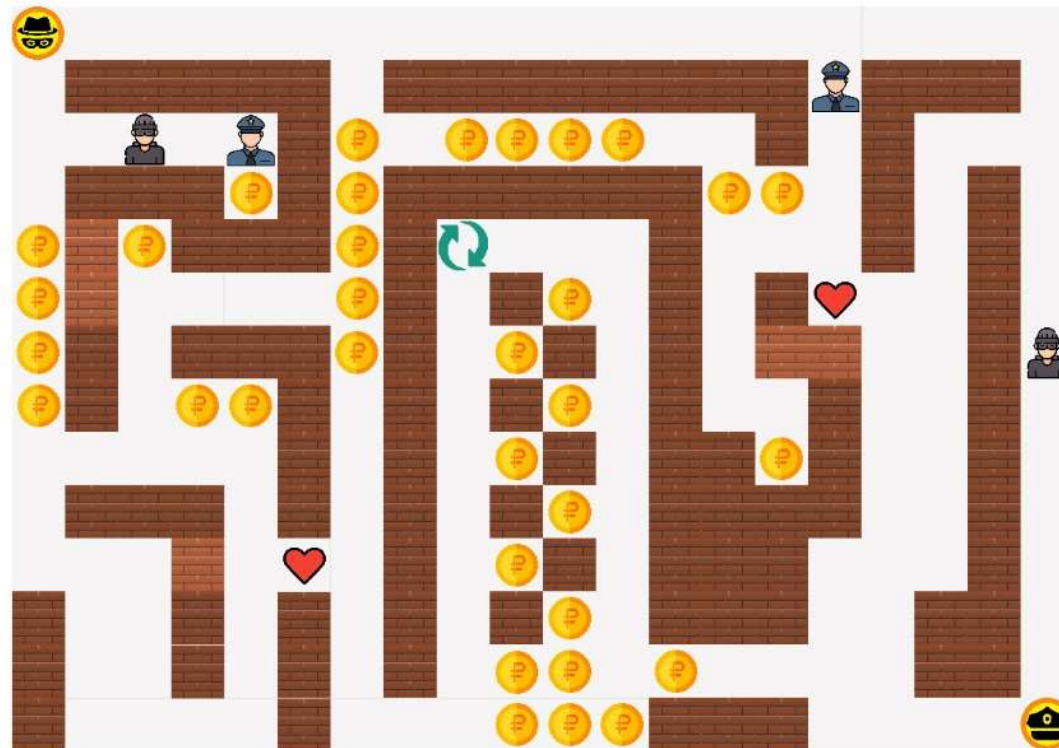
Точка возрождения жуликов

1. На экране 1 круговой индикатор при приближении счетчика таймера к нулю плавно меняет цвет до красного, за три секунды до 0 начинает мигать с изданием пищущего звука.
2. На экране 2 текст “смена команд” появляется одновременно со звуком ринга и показывается пользователю в течение трех секунд, затем заменяется таймером как на первом экране.

Рисунок 3 – Составные элементы лабиринта

## СМЕНА КОМАНД!

Команда 1  
125  
   



Команда 2  
131  
 

Рисунок 4 – Прототип сигнала о смене команд

## 4 Реализация интерфейса пользователя на HTML

### 4.1 Техническое задание

Сверстать страницу игры.

1) Создайте файл game.html. На странице должен быть canvas для отрисовки игры. Дать id для canvas, чтобы получить впоследствии доступ к нему из js;

2) На странице должны присутствовать кнопки для управления игрой: Старт, Стоп, Подключиться, Отключиться, Переподключение, Отменить игру, Выход. У каждой кнопки должен быть id;

3) Нужно добавить на страницу textarea для вывода логов и состояния игры. Дать свой id;

4) Нужно добавить div который впоследствии будет выполнять роль прогресс бара игры и будет показывать, сколько времени осталось до смены ролей. Дать id для div;

5) Нужно добавить два блока с названием команд, количеством очков и количеством оставшихся жизней. У блоков с названием, очками и жизнями должны быть id;

6) Требования к качеству выполнения задания:

6.1) семантическая верстка

6.2) отсутствие лишних элементов и блоков

6.3) прошло успешную проверку <https://validator.w3.org/> без ошибок

6.4) без табличной верстки

Итоговая верстка должна выглядеть так, как представлено на рисунке 5.

## Полицейские и мошенники

Смена ролей

полоса загрузки

Название: Синяки

Очки: 0

Осталось жизней: 5

Название: Курилки

Очки: 0

Осталось жизней: 5

Игра готова к запуску

Старт Стоп Подключиться Отключиться Переподключение Отменить игру Выход

Рисунок 5 – Образец верстки WEB-страницы

## 4.2 Реализованная WEB–страница

Листинг HTML–кода готовой WEB–страницы представлен ниже:

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>
      Полицейские и жулики
    </title>
    <link href="style.css" type="text/css" rel="stylesheet"/>
  </head>
  <body>
    <div class="gameWindow">
      <h1>
        Полицейские и жулики
      </h1>
      <div id="changeRoles">
        Смена ролей
        <div id="progressBarContainer">
          <div id="progressBarFill"></div>
        </div>
      </div>
      <div class="commandsContainer">
        <div id="commandFirst">
          <span class="commandNameText">Название:
</span><span id="nameFirst" class="commandName">Синяки</span><br>
          <span>Очки: </span><span
id="countFirst">0</span><br>
          <span>Осталось жизнью: </span><span
id="livesFirst">5</span><br>
        </div>
        <br>
        <div id="commandSecond">
          <span class="commandNameText">Название:
</span><span id="nameSecond" class="commandName">Курилки</span><br>
          <span>Очки: </span><span
id="countSecond">0</span><br>
          <span>Осталось жизнью: </span><span
id="livesSecond">5</span><br>
        </div>
      </div>
      <textarea id="logs">Игра готова к запуску</textarea><br>
      <canvas id="map" width="1090" height="602"></canvas><br>
      <div class="buttonsContainer">
        <button id="buttonStart"
class="buttonGreen">Старт</button>
        <button id="buttonStop"
class="buttonAqua">Стоп</button>
        <button id="buttonConnect"
class="buttonAqua">Подключиться</button>
```

```

        <button id="buttonDisconnect"
class="buttonAqua">Отключиться</button>
        <button
id="buttonReconnect" class="buttonAqua">Переподключение</button>
        <button id="buttonCancelGame"
class="buttonRed">Отменить игру</button>
        <button id="buttonExit"
class="buttonRed">Выход</button>
    </div>
</div>
</body>
</html>

```

Подготовленная страница отображается в браузере так, как представлено на рисунке 6.

## Полицейские и жулики

Смена ролей

Полоса загрузки

Название: Синяки

Очки: 0

Осталось жизней: 5

Название: Курилки

Очки: 0

Осталось жизней: 5

Игра готова к запуску

Старт

Стоп

Подключиться

Отключиться

Переподключение

Отменить игру

Выход

Рисунок 6 – Результат выполнения задания

## 5 Добавление к WEB–странице таблиц стилей CSS

### 5.1 Техническое задание

#### "Раскрасить" стилями страницу игры:

- 1) Создайте файл style.css и подключите его в game.html.
- 2) Центрирование страницы с игрой.
- 3) Установка фона с помощью CSS.
- 4) Вставка изображений <img> по желанию.
- 5) Размещение элементов <button>. При наведении мыши должен меняться курсор и цвет текста.
- 6) Расположить созданные элементы по странице в соответствии с личными предпочтениями (или хотя бы в соответствии с их предназначением).

#### Требования к качеству выполнения задания:

- 1) понятные названия классов (соответствуют элементу на странице)
- 2) использование абсолютного позиционирования (flex-box и/или grid по желанию)
- 3) без использования !important
- 4) без использования inline стилей

Итоговая верстка должна выглядеть так, как представлено на рисунке 7.

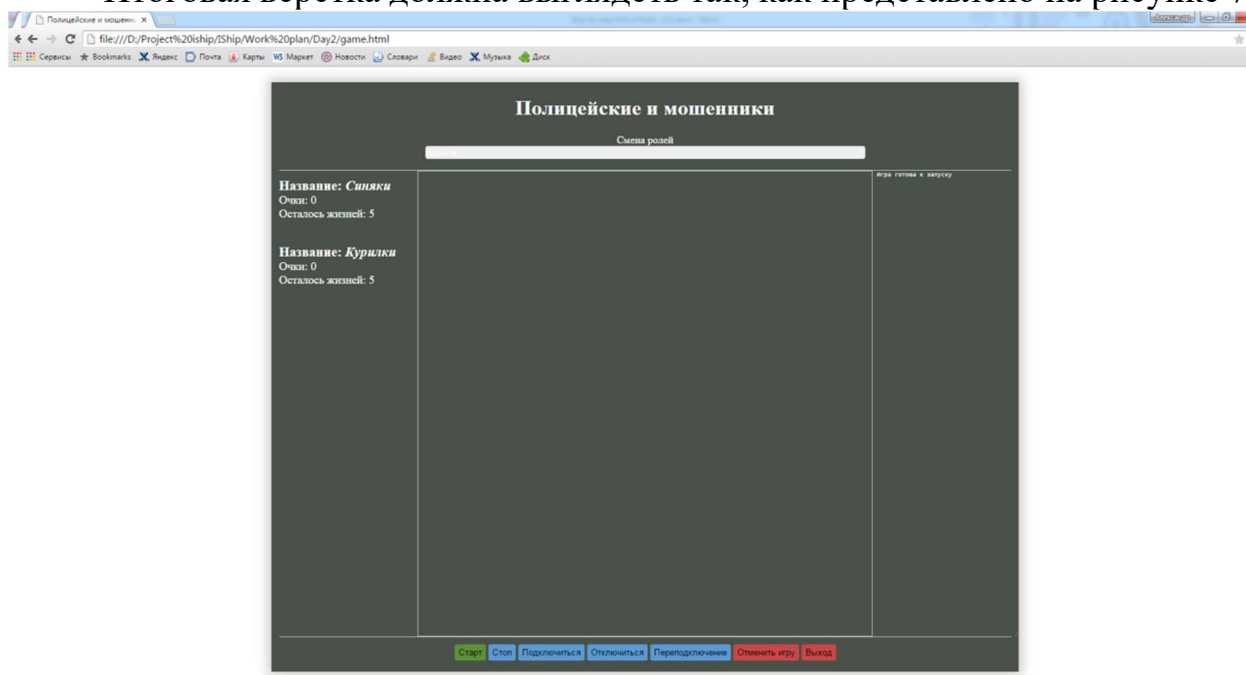


Рисунок 7 – Образец верстки WEB–страницы

### 5.2 Результат выполнения задания

Листинг CSS–стилей для WEB–страницы представлен ниже:

```
.gameWindow
{
    position: relative;
    width: 1500px;
    height: 800px;
    margin: auto;
    border: 10px solid rgb(91, 93, 83);
    background-color: rgb(91, 93, 83);
    color: white;
```



```

    h1
    {
        text-align: center;
    }
}

#changeRoles {
    position: absolute;
    width: 1500px;
    border-bottom: 2px solid white;
    padding: 5px 0px 15px 5px;
    margin: auto;
    text-align: center;
}

#progressBarContainer {
    width: 50%;
    margin: auto;
    background: white;
    text-align: left;
    border-radius: 3px;
    border: 1px solid #ccc;
    overflow: hidden;
    height: 15px;
}

#progressBarFill {
    width: 0%;
    height: 100%;
    background: red;
    border-radius: 3px 0 0 3px;
    transition: width 0.5s;
}

.commandsContainer
{
    position: absolute;
    top: 138px;
    width: 200px;
    height: 590px;
    margin: 5px;
}

#logs
{
    position: absolute;
    top: 138px;
    width: 200px;
    height: 590px;
    left: 1056px;
    resize: none;
}

```

```

border: none;
background-color: inherit;
color: inherit;
margin: 0px;
}

#map
{
    position: absolute;
    left: 446px;
    top: 138px;
    width: 602px;
    height: 602px;
    max-width: 1090px;
    max-height: 602px;
    background-color: inherit;
    border-right: 2px solid white;
    border-left: 2px solid white;
}

.buttonsContainer
{
    position: absolute;
    bottom: 0px;
    width: 1500px;
    height: 45px;
    border-top: 2px solid white;
    margin: auto;
    padding-top: 15px;
    text-align: center;
    button
    {
        border-radius: 5px;
        border: none;
        cursor: pointer;
        height: 25px;
    }

    button:hover {
        color: red;
    }

    .buttonGreen
    {
        background-color: green;
    }
    .buttonAqua
    {
        background-color: aqua;
    }
    .buttonRed

```

```

    {
        background-color: rgb(233, 104, 104);
    }
}

.commandNameText
{
    font-size: 20px;
    font-weight: bold;
}

.commandName
{
    font-size: 20px;
    font-weight: bold;
    font-style: italic;
}

```

Подготовленная страница отображается в браузере так, как представлено на рисунке 8.

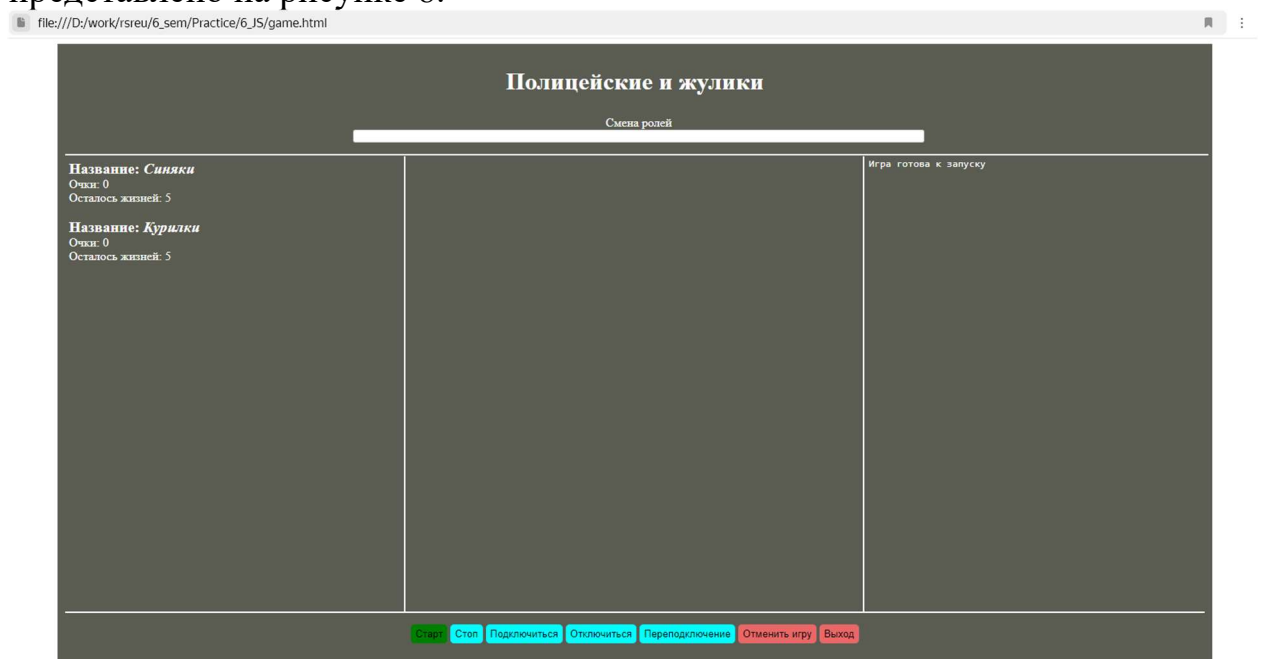


Рисунок 8 – WEB–страница с подключенными стилями

## 6 Реализация frontend–логики на языке JavaScript

### 6.1 Техническое задание

#### I) Подготовка:

Необходимо подключить все необходимые для работы скрипты:

```
<script src="../js/jquery.js" type="text/javascript"></script><!--required-->  
<script src="../js/jquery.signalR.js" type="text/javascript"></script><!--required-->  
<script src="../js/gameApi.js" type="text/javascript"></script><!--required-->  
<script src="jsHelper.js" type="text/javascript"></script>  
<script src="wrapper.js" type="text/javascript"></script>
```

Это скрипты, логику которых изменять не нужно, они нужны для работы игры. Внутри вашего скрипта необходимо создать экземпляр класса GameController `let gameController = new GameController(canvas)`. В качестве параметра, нужно передать объект `<canvas>` из html.

#### II) Реализовать базовую логику игры:

Для проверки логики в первую очередь необходимо зарегистрироваться на ресурсе <http://5.101.77.14:8085/>.

Залить обязательный список файлов для работы игры, попросите его у вашего ментора.

Теперь вам нужно реализовать три файла самостоятельно: `style.css`, `game.html`, `game.js`.

Первые два файла реализованы в предыдущих заданиях.

В `game.js` нужно разработать следующие функции:

- 1) `displayMap`;
- 2) `displayPlayers`;
- 3) `incrementProgress`;
- 4) `log`;
- 5) необходимо реализовать перемещение персонажа с помощью кнопок на клавиатуре.

III) Также необходимо написать обработчики всех кнопок. Методы которые будут использоваться в обработчиках, представлены в объекта `gameController`:

- 1) `start` - начать игру;
- 2) `stop` - остановить игру, пауза;
- 3) `cancel` - отменить игру, возобновить нельзя;
- 4) `reconnect` - переподключиться, нужно если с сетью проблемы;
- 5) `disconnect` - отключиться от игры, нужно нажать `reconnect`, чтобы вернуться `leave` - выйти из команды и стать подписчиком `join` - вернуться в команду (та же команда, очки и пр.).

Приветствуется, если скроете кнопки управления игрой, доступные только создателю этой игры. Для этого понадобится разработать класс `hidden` (к примеру), который будет навешиваться на кнопки, в случае если игрок не создатель игры.

После того как вы загрузили на сайт ваши файлы, необходимо нажать Перейти к списку игр, выбрать карту и создать игру. Если все было сделано правильно вы увидите работающую игру.

IV) Требования к качеству выполнения задания:

- 1) понятные названия функций и переменных;
- 2) работоспособный код без ошибок в консоли;
- 3) без использования var.

## 6.2 Реализация задания

Использовались материалы, представленные в источнике [2]. Листинг JavaScript-кода для WEB-страницы представлен ниже:

```
let cvs = document.getElementById('map');
let ctx = cvs.getContext('2d');
let gameController = new GameController(ctx);

function displayPlayers(players) {
  for (let i = 0; i < players.length; i++) {
    let p = players[i];
    let x = p.location.x * blockSizeX;
    let y = p.location.y * blockSizeY;
    ctx.drawImage(p.icon, x, y, blockSizeX, blockSizeY)
  }
}

function displayMap(map){
  if (map) {
    blockSizeX = cvs.clientWidth / map.width;
    blockSizeY = cvs.clientHeight / map.height;

    ctx.drawImage(gameController.icons.empty, 0, 0,
      cvs.clientWidth, cvs.clientHeight);

    if (ctx) {
      for (let i = 0; i < map.cells.length; i++) {
        let cell = map.cells[i];
        let x = parseInt(i/map.height);
        let y = i - x * map.height;
        displayCell(x, y, cell);
      }
    }
  }
}

gameController.incrementProgress = () =>{
  gameController.remainingSwitchTime += 100;
  setProgress(Math.floor(gameController.remainingSwitchTime /
    gameController.game.switchTimeout * 100));
  updateCoinsAndLives();
}

gameController.log = (message) =>{
  let textarea = document.getElementById('logs');
  textarea.value += message + '\n';
}
```

```
}
```

```
//Дополнительные функции и "костыли" из wrapper.js
```

```
function updateCoinsAndLives()
```

```
{
```

```
    let team1Coins = gameController.game.team1Stats.coinsCollected;
    let team1Lives = gameController.game.team1Stats.currentLives;
    let team2Coins = gameController.game.team2Stats.coinsCollected;
    let team2Lives = gameController.game.team2Stats.currentLives;
    document.getElementById('countFirst').innerText = team1Coins;
    document.getElementById('livesFirst').innerText = team1Lives;
    document.getElementById('countSecond').innerText = team2Coins;
    document.getElementById('livesSecond').innerText = team2Lives;
```

```
}
```

```
gameController.displayMap = (map) =>{
```

```
    displayMap(map);
```

```
}
```

```
gameController.displayPlayers = (players) =>{
```

```
    displayPlayers(players);
```

```
}
```

```
//Установка значений таймера
```

```
function setProgress(percentage) {
```

```
    if (percentage < 0) percentage = 0;
```

```
    if (percentage > 100) percentage = 100;
```

```
    let progressBarFill = document.getElementById('progressBarFill');
```

```
    progressBarFill.style.width = percentage + '%';
```

```
}
```

```
function defineCellType(type) {
```

```
    switch (type) {
```

```
        case GameApi.MapCellType.empty:
```

```
        default:
```

```
            return gameController.icons.empty || null;
```

```
        case GameApi.MapCellType.wall:
```

```
            return gameController.icons.wall || null;
```

```
        case GameApi.MapCellType.coin:
```

```
            return gameController.icons.coin || null;
```

```
        case GameApi.MapCellType.life:
```

```
            return gameController.icons.life || null;
```

```
        case GameApi.MapCellType.swtch:
```

```
            return gameController.icons.switch || null;
```

```
        case GameApi.MapCellType.thiefRespawn:
```

```
            return gameController.icons.empty || null;
```

```
        case GameApi.MapCellType.policeRespawn:
```

```
            return gameController.icons.empty || null;
```

```
        case 7:
```

```

        return gameController.icons.police || null;
    case 8:
        return gameController.icons.thief || null;
    }
}

function displayCell (x, y, type) {
    x *= blockSizeX;
    y *= blockSizeY;

    let img = defineCellType(type);
    if (img) {
        ctx.drawImage(img, x, y, blockSizeX, blockSizeY);
    }
};

function getCode(keyName) {
    switch(keyName) {
        case "ArrowLeft":
        case "a": // KeyA -> left
            return 37;
        case "ArrowUp":
        case "w": // KeyW -> up
            return 38;
        case "ArrowRight":
        case "d": // KeyD -> right
            return 39;
        case "ArrowDown":
        case "s": // KeyS -> down
            return 40;
        default:
            return -1;
    }
}

// Обработчики событий

document.addEventListener('keydown', (event) => {
    let keyCode = getCode(event.key);
    let direction = jsHelper.getDirection(keyCode);
    if (direction !== -1) {
        gameController.movePlayer(direction);
    }
});

let buttonStart = document.getElementById('buttonStart');
let buttonStop = document.getElementById('buttonStop');
let buttonConnect = document.getElementById('buttonConnect');
let buttonDisconnect = document.getElementById('buttonDisconnect');
let buttonReconnect = document.getElementById('buttonReconnect');
let buttonCancelGame = document.getElementById('buttonCancelGame');

```

```
let buttonExit = document.getElementById('buttonExit');
buttonStart.addEventListener('click', function() {
    updateCoinsAndLives()
    gameController.start();
});

buttonStop.addEventListener('click', function() {
    gameController.stop();
});

buttonConnect.addEventListener('click', function() {
    gameController.reconnect();
});

buttonDisconnect.addEventListener('click', function() {
    gameController.disconnect();
});

buttonReconnect.addEventListener('click', function() {
    gameController.join();
});

buttonCancelGame.addEventListener('click', function() {
    gameController.cancel();
});

buttonExit.addEventListener('click', function() {
    gameController.leave();
});
```



## 7 Тестирование серверной части WEB-приложения

### 7.1 Техническое задание

Используя swagger изучите доступные API методы для игры “Полицейские и грабители” по адресу <http://5.101.77.14:8085/swagger/>

Подготовьте postman коллекцию с запросами к каждому доступному методу и поделитесь ссылкой на нее. Методов всего 12.

Критерии приемки:

- 1) Коллекция доступна
- 2) У каждого запроса есть как минимум 2 скрипта-теста и один из них проверяет статус код
- 3) Коллекция запускается и тесты отрабатывают в зеленый
- 4) Поделиться либо ссылкой либо файлом. Если файлом – файл назвать имя\_фамилия\_город\_postman.json

Задание со звездочкой: вынести авторизационный токен в переменные и забирать его оттуда а не коипастить руками.

API для тестирования представлено на рисунке 9.

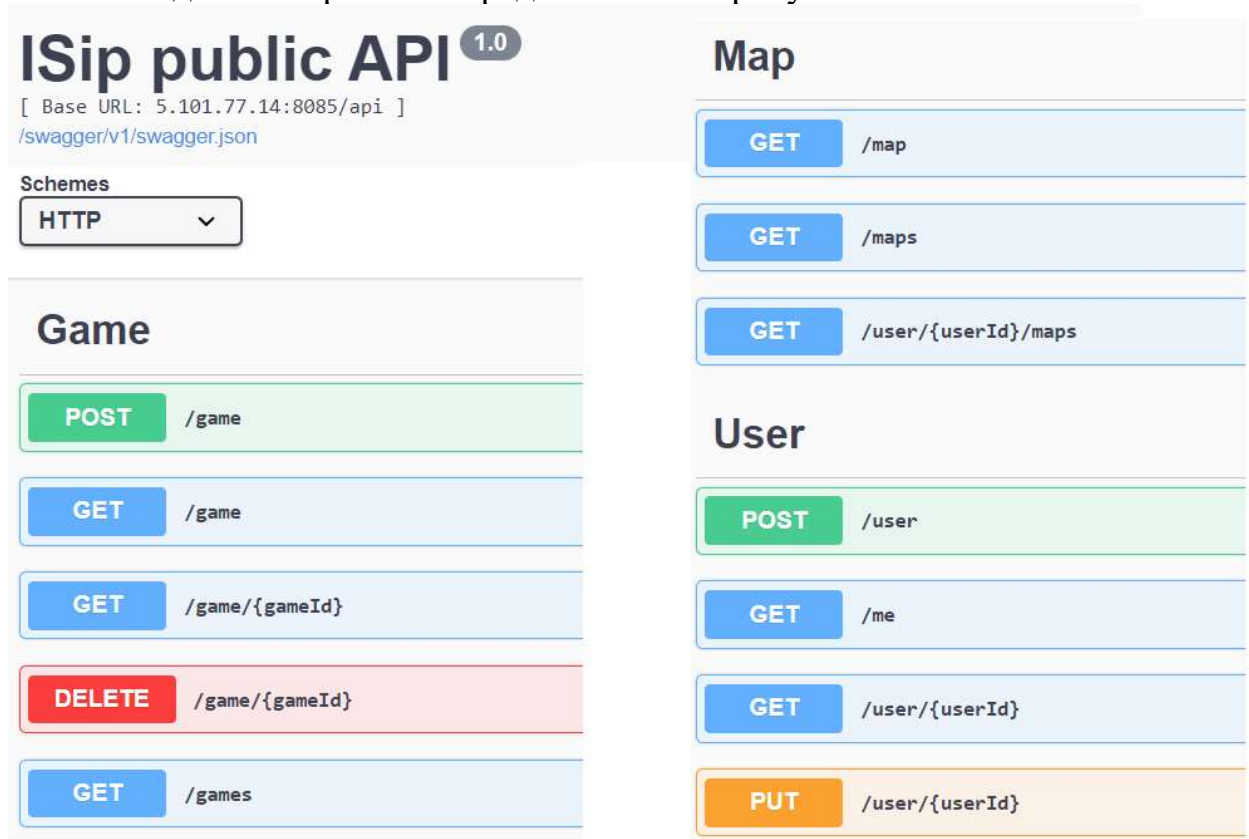


Рисунок 9 – API для тестирования

## 7.2 Результаты тестирования

Все написанные тесты завершились корректно. Более подробная информация о тестах (включая дополнительные требования к серверной части, например, максимальная допустимое время отклика системы, проверка кода ответа и т.д.) представлены на рисунках 10 и 11.

Erop\_Соколов\_Рязань\_Postman - Run results

Run Again

Automate Run ▾

Ran today at 14:55:49 · [View all runs](#)

Source	Environment	Iterations	Duration	All tests	Avg. Resp. Time
Runner	none	1	6s 971ms	25	395 ms

All Tests Passed (25) Failed (0) Skipped (0)

### Iteration 1

**POST** /game Создание игры

http://5.101.77.14:8085/api/game

200 OK 505 ms 907 B

PASS Ответ от сервера корректный

PASS Id есть в теле ответа

**GET** /game Получение информации о созданных играх

http://5.101.77.14:8085/api/game

200 OK 297 ms 2.074 KB

PASS Ответ от сервера корректен

PASS Ответ от сервера укладывается в 750 мс

**GET** /game/{gameId} Получение информации о игре по id

http://5.101.77.14:8085/api/game/8a337d331c874d149f517967c5da6add

200 OK 261 ms 33.979 KB

PASS Ответ от сервера корректен

PASS Ответ от сервера содержит поле id

PASS Имя карты корректно (testMap2000)

**DELETE** /game/{gameId} Удалить игру по id

http://5.101.77.14:8085/api/game/8a337d331c874d149f517967c5da6add

202 Accepted 87 ms 112 B

PASS Ответ от сервера корректен

PASS Ответ от сервера укладывается в 1000 мс

**GET** /games Получение информации о созданных играх

http://5.101.77.14:8085/api/games

200 OK 376 ms 1.321 KB

PASS Ответ от сервера корректен

PASS Ответ от сервера укладывается в 1000 мс

Рисунок 10 – Результаты тестирования, часть 1

<b>GET</b> /map Получить список всех карт	200 OK 158 ms 4.735 KB
http://5.101.77.14:8085/api/map	
PASS Ответ от сервера корректен	
PASS Ответ от сервера укладывается в 1250 мс	
<b>GET</b> /maps Получить список всех карт	200 OK 201 ms 4.735 KB
http://5.101.77.14:8085/api/maps	
PASS Ответ от сервера корректен	
PASS Ответ от сервера укладывается в 1250 мс	
<b>GET</b> /user/{id}/maps Получить список карт данного пользователя	200 OK 468 ms 155 B
http://5.101.77.14:8085/api/user/c8bf63a6207446b09729df9db43eed8c/maps	
PASS Ответ от сервера корректен	
PASS Ответ от сервера укладывается в 750 мс	
<b>POST</b> /user создание нового пользователя	201 Created 1079 ms 260 B
http://5.101.77.14:8085/api/user	
PASS Ответ от сервера корректен	
PASS Роль пользователя = 1	
<b>GET</b> /me Просмотр информации о своей учетной записи	200 OK 623 ms 263 B
http://5.101.77.14:8085/api/me	
PASS Ответ от сервера корректен	
PASS Роль пользователя != 0	
<b>GET</b> /user/{userId} Просмотр информации о учетной записи	200 OK 391 ms 255 B
http://5.101.77.14:8085/api/user/c1e6f25066694712a6adbdbbf79a61b6	
PASS Ответ от сервера корректен	
PASS Роль пользователя = 1	
<b>PUT</b> /user Обновление учетных данных	200 OK 293 ms 258 B
http://5.101.77.14:8085/api/user/c1e6f25066694712a6adbdbbf79a61b6	
PASS Ответ от сервера корректен	
PASS Время отклика меньше 1500 мс	

## Рисунок 11 – Результаты тестирования, часть 2

При подготовке тестов в системе Postman использовалось справочное руководство [3].

## 8 Анализ разработанного приложения при помощи практик Business Intelligence

### 8.2 Техническое задание

Вы создали игру, которая становится популярной и в нее начинают играть во многих странах. Заказчик хотел бы ежедневно анализировать статистику по игрокам, чтобы своевременно принимать бизнес-решения. Ваша задача – создать дашборд, который поможет ответить на вопросы:

- 1) Общее количество уникальных игроков
- 2) Количество игроков по странам
- 3) Динамика популярности игры по месяцам
- 4) Пользователи из какого города чаще всего играют в игру.
- 5) Топ игроков, которые провели больше всего времени в игре.
- 6) В скольких городах Австралии играют в игру
- 7) Количество игроков, которые не соответствуют нашим ожиданиям по времени в игре
- 8) Какая доля российских игроков

#### Детали:

Структура базы данных представлена на рисунке 12.

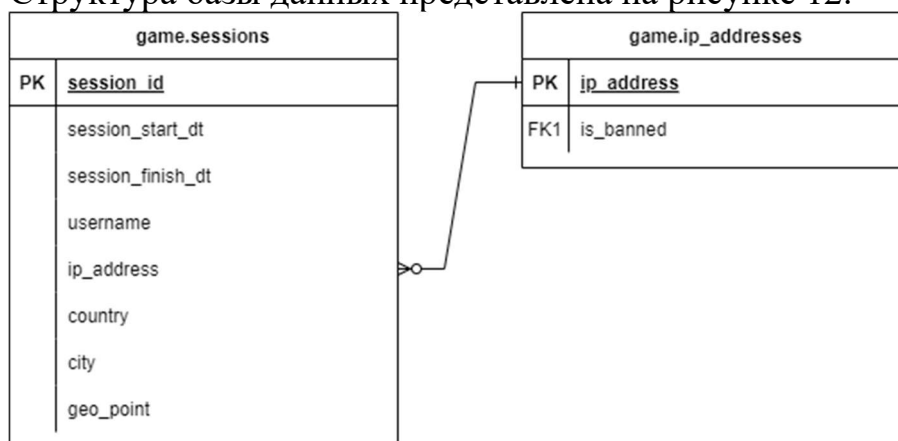


Рисунок 12 – Структура базы данных

#### Работа с базой данных:

- 1) Скачайте репозиторий <https://github.com/sergeiboikov/Summer.Practice.BI>;
- 2) Создайте структуру базы данных в Postgres, выполнив SQL запрос «SQL\1\_Create DB structure.sql»;
- 3) В DBeaver создайте задачу «game.sessions import» для импорта данных из CSV файла «CSV\sessions\_data.csv» в таблицу «game.sessions»;
- 4) Используя задачу из предыдущего шага, импортируйте данные из файла «CSV\sessions\_data.csv» в таблицу «game.sessions»;
- 5) Выполните запрос «SQL\2\_Populate data.sql» для вставки дополнительных данных;
- 6) Создайте представление «dm.v\_sessions» с полями, представленными в таблице 1.

Таблица 1 –Описание полей представления «dm.v\_sessions»

Название поля	Описание
session_id	ID сессии из таблицы «game.sessions»
Session Start	Дата и время начала сессии
Session Finish	Дата и время окончания сессии
Session Duration (sec)	Расчетное поле с длительностью сессии в секундах. Можете использовать для этого функции «EXTRACT» и «EPOCH»
User	Пользователь из таблицы «game.sessions»
Country	Страна из таблицы «game.sessions»
City	Город из таблицы «game.sessions»
Geo Point	Гео точка из таблицы «game.sessions»

**Важно:** В представление «dm.v\_sessions» не должны попадать игроки с забаненными IP адресами. Для этого добавьте фильтр, используя таблицу «game.ip\_addresses».

**Примечание:** SQL запрос представления вам понадобится дальше в одном из вопросов при заполнении Yandex Forms. В DBeaver создайте задачу «dm.v\_sessions export» для экспорта данных из представления «dm.v\_sessions» в CSV файл.

#### **Работа с Yandex Datalens:**

- 1) В Datalens создайте подключение к файлу
- 2) Создайте датасет «Sessions dataset»
- 3) Создайте в датасете рассчитываемое поле «Session Duration (hours)» с длительностью сессии в часах.
- 4) Создайте чарт с типом «Индикатор», показывающий общее число игроков.
- 5) Создайте иерархию «Geography» и включите в нее поля «Country» и «City».
- 6) Создайте чарт с типом «Круговая диаграмма», показывающий количество уникальных игроков по иерархии «Geography».
- 7) Создайте чарт с типом «Линейная диаграмма», показывающий динамику количества сессий по месяцам (для дат используйте поле «Session Start»).
- 8) Создайте чарт с типом «Карта», показывающий количество сессий по городам.
- 9) Создайте чарт с типом «Столбчатая диаграмма», показывающий длительность сессий по игрокам.
- 10) Создайте любые чарты, которые посчитаете нужными.
- 11) Создайте дашборд.
- 12) Добавьте на дашборд созданные ранее чарты.
- 13) Добавьте на дашборд необходимые селекторы.

14) Откройте публичный доступ к дашборду. Ссылку необходимо будет указать далее в Yandex Forms

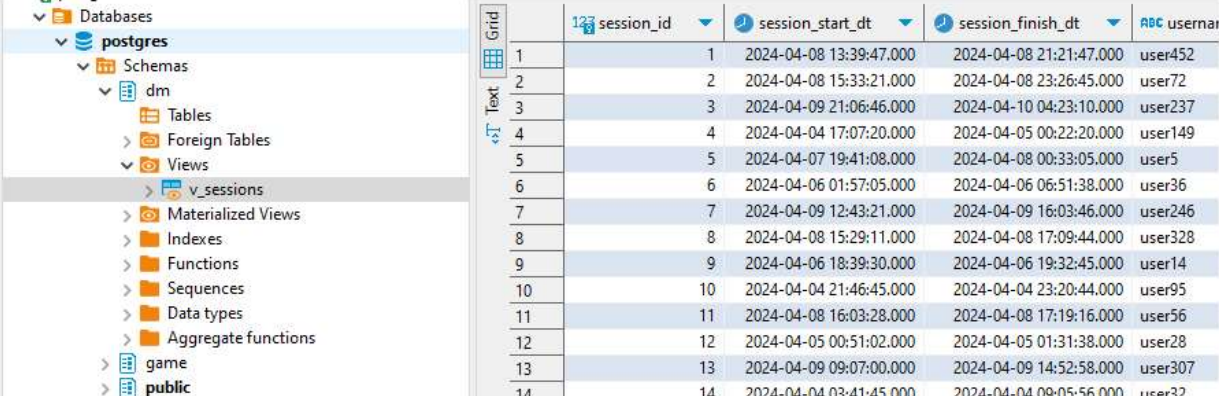
15) Используя дашборд, ответьте на вопросы из Yandex Forms по ссылке <https://forms.yandex.ru/cloud/66682dd5f47e732e90f75ca1/>

## 8.2 Результат выполнения задания

Требуемые таблицы БД были созданы и заполнены при помощи скриптов и CSV файла, присланными организаторами практики. Листинг написанного запроса для создания представления «dm.v\_sessions» помещен ниже.

```
CREATE OR REPLACE VIEW dm.v_sessions
AS SELECT
    game.sessions.session_id as "session_id",
    game.sessions.session_start_dt AS "Session Start",
    game.sessions.session_finish_dt AS "Session Finish",
    EXTRACT(EPOCH FROM (game.sessions.session_finish_dt -
game.sessions.session_start_dt)) AS "Session Duration (sec)",
    game.sessions.username AS "User",
    game.sessions.country AS "Country",
    game.sessions.city AS "City",
    game.sessions.geo_point AS "Geo point"
FROM game.sessions
WHERE
    NOT EXISTS (
        SELECT 1
        FROM game.ip_addresses AS ip
        WHERE ip.ip_address = game.sessions.ip_address
        AND ip.is_banned = TRUE
    );
```

Результат выполнения запроса представлен на рисунке 13.



	123 session_id	session_start_dt	session_finish_dt	ABC username
1	1	2024-04-08 13:39:47.000	2024-04-08 21:21:47.000	user452
2	2	2024-04-08 15:33:21.000	2024-04-08 23:26:45.000	user72
3	3	2024-04-09 21:06:46.000	2024-04-10 04:23:10.000	user237
4	4	2024-04-04 17:07:20.000	2024-04-05 00:22:20.000	user149
5	5	2024-04-07 19:41:08.000	2024-04-08 00:33:05.000	user5
6	6	2024-04-06 01:57:05.000	2024-04-06 06:51:38.000	user36
7	7	2024-04-09 12:43:21.000	2024-04-09 16:03:46.000	user246
8	8	2024-04-08 15:29:11.000	2024-04-08 17:09:44.000	user328
9	9	2024-04-06 18:39:30.000	2024-04-06 19:32:45.000	user14
10	10	2024-04-04 21:46:45.000	2024-04-04 23:20:44.000	user95
11	11	2024-04-08 16:03:28.000	2024-04-08 17:19:16.000	user56
12	12	2024-04-05 00:51:02.000	2024-04-05 01:31:38.000	user28
13	13	2024-04-09 09:07:00.000	2024-04-09 14:52:58.000	user307
14	14	2024-04-04 03:41:45.000	2024-04-04 09:05:56.000	user32

Рисунок 13 – Результат выполнения запроса

Вид итогового дашборда в облаке Yandex представлен на рисунке 14.

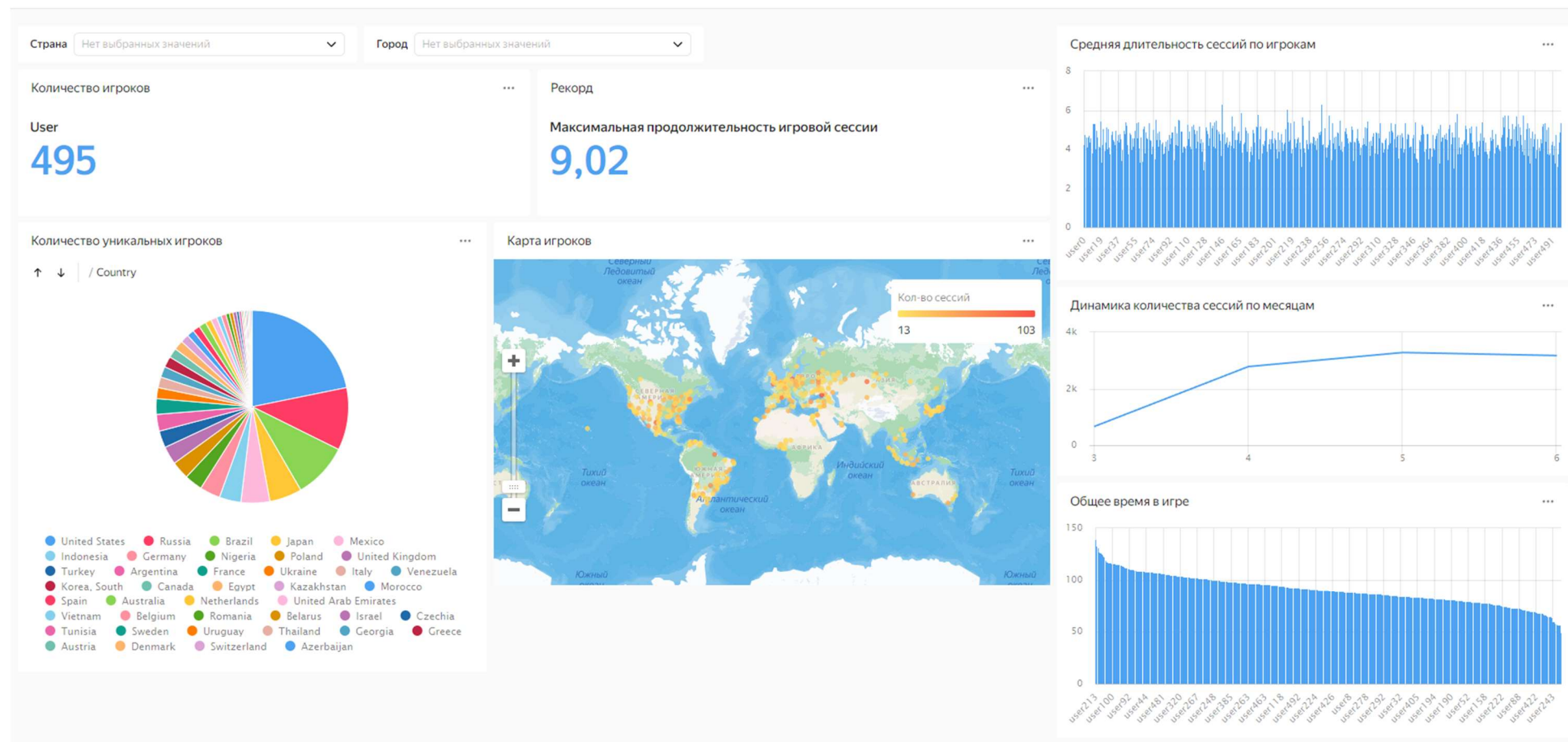


Рисунок 14 – Итоговый дашборд



## 9 Дополнительные лекции

В течение практики также были проведены дополнительные лекции (без домашних заданий) по:

1) **Бэкенду**, где было дано поверхностное описание принципов работы клиент–серверных приложений в общем и предоставлена часть исходного кода серверной части разрабатываемого приложения в частности;

2) **DevOps**, где было дано введение в основные практики данного направления и описаны основные инструменты для автоматизации и управления инфраструктурой. Были упомянуты:

2.1) Контейнеризация (Docker) и принципы работы с контейнерами.

2.2) CI/CD: Введение в непрерывную интеграцию и непрерывное развёртывание.

2.3) Необходимые навыки для успешной работы DevOps–специалистом.

3) **Soft Skills**, где основное внимание было уделено стратегиям по развитию навыка стрессоустойчивости. Обсуждались следующие темы:

3.1) Управление временем: Техники планирования и приоритизации задач для снижения уровня стресса;

3.2) Методы релаксации: Практические упражнения по медитации и дыхательным техникам для снижения напряжения;

3.3) Эмоциональный интеллект: Развитие навыков по распознаванию и управлению своими эмоциями;

3.4) Постановка целей и мотивация: Техники постановки достижимых целей и поддержания мотивации в условиях стресса.



## ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ ПРАКТИКИ

Результаты прохождения практики должны быть представлены с указанием достигнутого уровня компетенций, представленных в таблице 2

Таблица 2

Категория (группа) универсальных компетенций	Код и наименование универсальной компетенции	Код и наименование индикатора достижения универсальной компетенции
Системное критическое мышление	и УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач.	ИД – 1 УК-1 Знать: принципы сбора, отбора и обобщения информации, методики системного подхода для решения профессиональных задач. ИД – 2 УК-1 Уметь: анализировать и систематизировать разнородные данные, оценивать эффективность процедур анализа проблем и принятия решений в профессиональной деятельности. ИД – 3 УК-1 Владеть: навыками научного поиска и практической работы с информационными источниками; методами принятия решений.
Разработка и реализация проектов	и УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	ИД-1 УК-2 Знает необходимые для осуществления профессиональной деятельности правовые нормы и методологические основы принятия управленческого решения. ИД-2 УК-2 Умеет анализировать альтернативные варианты решений для достижения намеченных результатов; разрабатывать план, определять целевые этапы и основные направления работ. ИД-3 УК-2 Владеет методиками разработки цели и задач проекта; методами оценки продолжительности и стоимости проекта, а также потребности в ресурсах
Самоорганизация и саморазвитие (в том числе здоровьесбережение)	и УК-6. Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни	ИД-1 УК-6 Знает основные принципы самовоспитания и самообразования, исходя из требований рынка труда. ИД-2 УК-6

		<p>Умеет демонстрировать умение самоконтроля и рефлексии, позволяющие самостоятельно корректировать обучение по выбранной траектории.</p> <p>ИД-3 ук-6</p> <p>Владеет способами управления своей познавательной деятельностью и удовлетворения образовательных интересов и потребностей</p>
Безопасность жизнедеятельности	<p>УК-8. Способен создавать и поддерживать безопасные условия жизнедеятельности, в том числе при возникновении чрезвычайных ситуаций</p>	<p>ИД-1 ук-8</p> <p>Знает причины, признаки и последствия опасностей, способы защиты от чрезвычайных ситуаций; основы безопасности жизнедеятельности, телефоны служб спасения.</p> <p>ИД-2 ук-8</p> <p>Умеет выявлять признаки, причины и условия возникновения чрезвычайных ситуаций; оценивать вероятность возникновения потенциальной опасности для обучающегося и принимать меры по ее предупреждению в условиях образовательного учреждения; оказывать первую помощь в чрезвычайных ситуациях</p>

### 1) Вводная лекция и инструктаж по технике безопасности:

- УК-8. Способен создавать и поддерживать безопасные условия жизнедеятельности в том числе при возникновении чрезвычайных ситуаций.
- ИД-1 УК-8: Знает причины, признаки и последствия опасностей; способы защиты от чрезвычайных ситуаций; основы безопасности жизнедеятельности; телефоны служб спасения.
- ИД-2 УК-8: Умеет выявлять признаки, причины и условия возникновения чрезвычайных ситуаций; оценивать вероятность возникновения потенциальной опасности и принимать меры по её предупреждению; оказывать первую помощь в чрезвычайных ситуациях.

### 2) Лекция по методикам проектирования:

- УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач.

- ИД-1 УК-1: Знать принципы сбора, отбора и обобщения информации; методики системного подхода для решения профессиональных задач.
- ИД-2 УК-1: Уметь анализировать и систематизировать разнородные данные; оценивать эффективность процедур анализа проблем и принятия решений в профессиональной деятельности.
- ИД-3 УК-1: Владеть навыками научного поиска и практической работы с информационными источниками; методами принятия решений.

### **3) Бизнес и системный анализ:**

- УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения исходя из действующих правовых норм, имеющихся ресурсов и ограничений.
- ИД-1 УК-2: Знает необходимые для осуществления профессиональной деятельности правовые нормы и методологические основы принятия управленческого решения.
- ИД-2 УК-2: Умеет анализировать альтернативные варианты решений для достижения намеченных результатов; разрабатывать план, определять целевые этапы и основные направления работ.
- ИД-3 УК-2: Владеть методиками разработки цели и задач проекта; методами оценки продолжительности и стоимости проекта, а также потребности в ресурсах.

### **4) Проектирование макета интерфейса пользователя:**

- УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения исходя из действующих правовых норм, имеющихся ресурсов и ограничений.
- ИД-2 УК-2: Умеет анализировать альтернативные варианты решений для достижения намеченных результатов; разрабатывать план, определять целевые этапы и основные направления работ.
- ИД-3 УК-2: Владеть методиками разработки цели и задач проекта; методами оценки продолжительности и стоимости проекта, а также потребности в ресурсах.

### **5) Лекция по серверной части клиент–серверных приложений:**

- УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач.
- ИД-1 УК-1: Знать принципы сбора, отбора и обобщения информации; методики системного подхода для решения профессиональных задач.
- ИД-2 УК-1: Уметь анализировать и систематизировать разнородные данные; оценивать эффективность процедур анализа проблем и принятия решений в профессиональной деятельности.

### **6) Реализация фронтенд–части приложения на HTML + CSS + JavaScript:**

- УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения исходя из действующих правовых норм, имеющихся ресурсов и ограничений.

- ## 7) Тестирование серверной части приложения:

- ## 8) Анализ разработанного приложения при помощи практик VI:

- 9) Лекция по Soft Skills (стрессоустойчивость):**

- Студент Соколов Е.А. / (Ф.И.О., подпись)

Руководитель практики от предприятия Силкин Г.Д. / подпись  
(Ф.И.О.,

## **БИБЛИОГРАФИЧЕСКИЙ СПИСОК**

- 1 Интернет-ресурс [www.figma.com](https://www.figma.com/design/) – URL: <https://www.figma.com/design/> (дата обращения: 01.07.2024);
- 2 В.В. Дунаев, JavaScript. 3–е изд. – СПб.: Питер, 2008 – 400 с.;
- 3 Интернет-ресурс [www.habr.com](https://habr.com/ru/companies/vk/articles/750096/) – URL: <https://habr.com/ru/companies/vk/articles/750096/> (дата обращения: 09.07.2024);

## ОТЗЫВ

### РУКОВОДИТЕЛЯ ПРАКТИКИ ОТ ПРЕДПРИЯТИЯ

**Студент:** Соколов Егор Александрович

**Группа:** 143, специальность 09.03.04 «Программная инженерия», РГРТУ

Проходивший учебную практику с «21» июня 2024г. по «18» июля 2024г.

**Место прохождения практики:** ООО «РНТ»

За время прохождения практики студент продемонстрировал высокую способность к самостоятельной работе, получил навыки анализа, моделирования и работы с современными информационными системами, а также навыки разработки промышленного ПО на языках высокого уровня: C#, JavaScript. Также отдельно стоит отметить внимательность, прилежание, серьезное отношение к работе и опрятный внешний вид.

#### **Приобрел практический опыт:**

Участие в выработке требований к программному обеспечению

Участие в проектировании программного обеспечения с использованием специализированных программных пакетов

Разработки и тестирования программного обеспечения; с основами анализа, выявления и требований к информационным системам; с моделями и жизненным циклом разработки информационных систем; с разработкой web и desktop приложений; с основами тестирования и обеспечения качества информационных систем.

#### **Приобрел умения:**

Владеть основными методологиями процессов разработки ПО;

Использовать методы для получения кода с заданной функциональностью и степенью качества;

Грамотно и качественно писать код на языке высокого уровня.

#### **Выводы, рекомендации:**

Практику можно считать успешно пройденной, студент заслуживает оценки «отлично».

Руководитель практики от организации: Силкин Григорий Дмитриевич.

Руководитель практики от предприятия: \_\_\_\_\_

Силкин Г.Д.

(Ф.И.О.,

подпись)

МП  
(печать предприятия)

