**1. Добавление новой игры: "Сапер с элементами матч-3"**

**1.1 Описание игры**

* Игра представляет собой гибрид классического "Сапера" и механики "матч-3".
* Игровое поле состоит из клеток, каждая из которых может содержать:
  + Число (количество мин вокруг клетки).
  + Мину (клетка, которую нельзя открывать).
  + Специальный спрайт (персонаж или предмет).

**1.2 Механика игры**

1. **Основная механика "Сапера"**:
   * Пользователь открывает клетки, пытаясь избежать мин.
   * Если пользователь открывает клетку с миной, игра заканчивается.
   * Если пользователь открывает клетку с числом, оно показывает, сколько мин находится в соседних клетках.
2. **Механика "матч-3"**:
   * Если пользователь открывает три клетки с одинаковыми спрайтами (персонажами или предметами), эти спрайты исчезают, и пользователь получает монеты.
   * Исчезнувшие спрайты добавляются в специальный столбец (инвентарь), где они могут быть использованы для дальнейших бонусов или улучшений.
3. **Награды**:
   * За каждые три собранных одинаковых спрайта пользователь получает монеты.
   * Монеты могут быть использованы для покупки подсказок, улучшений или участия в других играх (лотерея, казино).

**1.3 Интерфейс игры**

* Игровое поле отображается в виде сетки (например, 8x8 или 10x10 клеток).
* В верхней части экрана отображается количество монет, заработанных в игре.
* В нижней части экрана отображается инвентарь с собранными спрайтами.

**2. Дополнительные функции**

**2.1 Улучшения и бонусы**

* Пользователь может использовать собранные спрайты для получения бонусов:
  + **Подсказка**: Показывает, где находится ближайшая мина.
  + **Защита**: Позволяет открыть одну клетку с миной без проигрыша.
  + **Увеличение награды**: Увеличивает количество монет за собранные спрайты.

**2.2 Лидерборд**

* Добавление таблицы лидеров, где пользователи могут сравнивать свои результаты в игре.
* Лидерборд обновляется в реальном времени и отображает топ-10 игроков.

**2.3 Ежедневные задания**

* Добавление ежедневных заданий, связанных с игрой:
  + "Открой 10 клеток без мин".
  + "Собери 5 наборов из трех одинаковых спрайтов".
  + "Заработай 100 монет в игре".

**3. Техническая реализация**

**3.1 Микросервис для игры**

* **Game Service**:
  + Отвечает за логику игры "Сапер с элементами матч-3".
  + Генерация игрового поля.
  + Обработка действий пользователя (открытие клеток, сбор спрайтов).
  + Начисление монет и бонусов.

**3.2 Взаимодействие с другими сервисами**

* **User Service**:
  + Обновление баланса пользователя (монеты, бонусы).
* **Task Service**:
  + Проверка выполнения ежедневных заданий, связанных с игрой.
* **Leaderboard Service**:
  + Обновление таблицы лидеров на основе результатов игры.

**3.3 Хранение данных**

* **PostgreSQL**:
  + Хранение информации о игровых сессиях, собранных спрайтах и наградах.
* **Redis**:
  + Кэширование данных для быстрого доступа к информации о текущей игровой сессии.

**4. Этапы разработки игры**

1. **Проектирование**:
   * Определение механики игры.
   * Проектирование API для взаимодействия с другими сервисами.
2. **Разработка**:
   * Реализация логики игры.
   * Интеграция с User Service и Task Service.
3. **Тестирование**:
   * Модульное тестирование игровой логики.
   * Интеграционное тестирование с другими сервисами.
4. **Деплой**:
   * Развертывание Game Service в Kubernetes.
   * Настройка мониторинга и логирования.
5. **Поддержка**:
   * Сбор обратной связи от пользователей.
   * Внесение улучшений и исправление багов.

**2. Роли и их возможности**

**2.1 Пользователь**

* **Авторизация через Telegram**:
  + Пользователь авторизуется через Telegram, используя Telegram API.
  + После авторизации пользователь получает доступ к своему профилю и функционалу приложения.
* **Реферальная программа**:
  + Пользователь может создать реферальную ссылку для приглашения друзей.
  + Бонусы начисляются только при достижении определенного количества очков приглашенными пользователями.
  + Пользователь может отслеживать количество приглашенных друзей и их активность.
* **Задания (Tasks)**:
  + Пользователь может просматривать список заданий.
  + За выполнение заданий пользователь получает награды (очки, бонусы и т.д.).
  + Задания могут быть ежедневными, еженедельными или одноразовыми.
* **Дополнительные функции**:
  + Пользователь может перейти на страницу с кнопками: Market, Lottery, Casino, Staking.
  + На данный момент доступны только функции Lottery и Casino:
    - **Lottery**: Участие в лотерее с возможностью выигрыша призов.
    - **Casino**: Игры в казино с возможностью выигрыша очков или бонусов.

**2.2 Администратор**

* **Управление пользователями**:
  + Просмотр списка пользователей.
  + Блокировка/разблокировка пользователей.
  + Назначение бонусов или штрафов.
* **Управление заданиями**:
  + Создание, редактирование и удаление заданий.
  + Настройка наград за выполнение заданий.
* **Управление реферальной программой**:
  + Настройка условий для получения бонусов.
  + Просмотр статистики по реферальным ссылкам.
* **Управление функциями Lottery и Casino**:
  + Настройка правил и условий для лотереи и казино.
  + Просмотр статистики по играм.

**3. Микросервисная архитектура**

**3.1 Сервисы**

1. **Auth Service**:
   * Отвечает за авторизацию пользователей через Telegram.
   * Генерация и валидация JWT-токенов.
2. **User Service**:
   * Управление профилями пользователей.
   * Хранение информации о пользователях (ID, имя, реферальные ссылки, бонусы и т.д.).
3. **Referral Service**:
   * Управление реферальной программой.
   * Генерация реферальных ссылок.
   * Начисление бонусов за приглашенных пользователей.
4. **Task Service**:
   * Управление заданиями.
   * Начисление наград за выполнение заданий.
5. **Lottery Service**:
   * Управление лотереей.
   * Проведение розыгрышей и начисление призов.
6. **Casino Service**:
   * Управление играми в казино.
   * Начисление выигрышей.
7. **Admin Service**:
   * Управление административными функциями.
   * Взаимодействие с другими сервисами для получения статистики и управления данными.

**3.2 Взаимодействие сервисов**

* Сервисы взаимодействуют друг с другом через HTTP API или gRPC.
* Для хранения данных используется база данных PostgreSQL.
* Для кэширования данных используется Redis.
* Для асинхронных задач (например, начисление бонусов) используется RabbitMQ или Kafka.

### Техническое задание (ТЗ) на дизайн интерфейса в стиле киберпанк

#### ****1. Общие требования****

##### ****1.1 Стилистика****

* **Киберпанк**: Основная стилистика интерфейса должна быть выдержана в стиле киберпанк с использованием неоновых цветов, технологичных элементов и урбанистических мотивов.
* **Футуристический город**: На фоне интерфейса должен быть изображен заполненный футуристический город с небоскребами, летающими машинами, неоновыми вывесками и другими элементами, характерными для киберпанка.

##### ****1.2 Цветовая палитра****

* Основные цвета: неоново-синий, фиолетовый, розовый, зеленый.
* Дополнительные цвета: черный, серый, белый для контраста.
* Акценты: яркие неоновые элементы для выделения важных элементов интерфейса.

##### ****1.3 Шрифты****

* Основной шрифт: техно-шрифт с геометрическими формами (например, "Exo 2" или "Orbitron").
* Дополнительный шрифт: моноширинный шрифт для отображения технической информации (например, "Roboto Mono").

#### ****2. Элементы интерфейса****

##### ****2.1 Главный экран****

* **Фон**: Футуристический город с небоскребами, летающими машинами и неоновыми вывесками.
* **Кнопки**:
  + **Referral**: Неоновая кнопка с иконкой ссылки.
  + **Tasks**: Неоновая кнопка с иконкой списка заданий.
  + **App**: Неоновая кнопка с иконкой приложения.
  + **Game**: Неоновая кнопка с иконкой игры (например, значок мины или здания).

##### ****2.2 Экран реферальной программы****

* **Фон**: Часть города с неоновыми вывесками и рекламными баннерами.
* **Элементы**:
  + Поле для создания реферальной ссылки.
  + Список приглашенных друзей с их активностью.
  + Прогресс-бар для отслеживания достижения бонусов.

##### ****2.3 Экран заданий (Tasks)****

* **Фон**: Узкая улица с неоновыми вывесками и кибер-граффити.
* **Элементы**:
  + Список заданий с иконками и описанием.
  + Прогресс выполнения заданий.
  + Кнопка "Claim" для получения награды.

##### ****2.4 Экран приложения (App)****

* **Фон**: Вид на город с высоты, с летающими машинами и небоскребами.
* **Кнопки**:
  + **Market**: Неоновая кнопка с иконкой магазина.(Временно неактивна)
  + **Lottery**: Неоновая кнопка с иконкой лотереи.
  + **Casino**: Неоновая кнопка с иконкой казино.
  + **Staking**: Неоновая кнопка с иконкой стекинга (временно неактивна).

##### ****2.5 Экран игры "Сапер с элементами матч-3"****

* **Фон**: Футуристический город с многоэтажками, которые заменяют клетки игрового поля.
* **Элементы**:
  + **Игровое поле**: Многоэтажки разной высоты и стиля, каждая из которых представляет собой клетку.
  + **Спрайты**: Неоновые персонажи или предметы, которые появляются на зданиях.
  + **Инвентарь**: Столбец в правой части экрана, где отображаются собранные спрайты.
  + **Счетчик монет**: В верхней части экрана, отображает количество заработанных монет.

#### ****3. Детализация дизайна****

##### ****3.1 Игровое поле****

* **Многоэтажки**: Каждая клетка игрового поля представлена в виде многоэтажки с неоновыми окнами и вывесками.
* **Анимации**:
  + При открытии клетки (многоэтажки) появляется анимация подсветки здания.
  + При сборе трех одинаковых спрайтов здания "взрываются" с неоновыми эффектами.

##### ****3.2 Спрайты****

* **Персонажи**: Неоновые персонажи в стиле киберпанк (например, хакеры, роботы, киборги).
* **Предметы**: Технологичные предметы (например, чипы, батарейки, гаджеты).

##### ****3.3 Инвентарь****

* **Столбец**: Вертикальный столбец в правой части экрана, где отображаются собранные спрайты.
* **Анимации**: При добавлении нового спрайта в инвентарь появляется анимация "всплывания".

##### ****3.4 Навигация****

* **Меню**: В нижней части экрана расположено меню с иконками для быстрой навигации между разделами приложения.
* **Анимации переходов**: При переходе между экранами используются анимации с неоновыми эффектами (например, "проявление" нового экрана).

#### ****4. Требования к анимациям и эффектам****

##### ****4.1 Анимации****

* **Открытие клеток**: Подсветка здания неоновым цветом.
* **Сбор спрайтов**: Взрыв здания с неоновыми частицами.
* **Переходы между экранами**: Плавное появление и исчезновение элементов с неоновыми эффектами.

##### ****4.2 Эффекты****

* **Неоновое свечение**: Все кнопки, иконки и важные элементы интерфейса должны иметь неоновое свечение.
* **Параллакс-эффект**: На фоне игрового поля может быть использован параллакс-эффект для создания ощущения глубины.

#### ****5. Этапы разработки дизайна****

1. **Исследование и концепция**:
   * Сбор референсов по стилю киберпанк.
   * Разработка концепции интерфейса.
2. **Прототипирование**:
   * Создание wireframes основных экранов.
   * Утверждение структуры интерфейса.
3. **Дизайн**:
   * Разработка дизайна всех экранов в соответствии с утвержденной концепцией.
   * Создание анимаций и эффектов.
4. **Тестирование**:
   * Проверка дизайна на удобство использования.
   * Внесение корректировок на основе обратной связи.
5. **Реализация**:
   * Передача дизайна разработчикам.
   * Поддержка в процессе реализации.

#### ****6. Заключение****

Дизайн интерфейса в стиле киберпанк с футуристическим городом на фоне и многоэтажками вместо клеток создаст уникальную атмосферу для приложения. Использование неоновых цветов, технологичных элементов и анимаций сделает интерфейс привлекательным и запоминающимся для пользователей.