



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

КУРСОВАЯ РАБОТА

по дисциплине: Разработка серверных частей интернет-ресурсов
по профилю: Разработка и дизайн компьютерных игр и мультимедийных приложений
направления профессиональной подготовки: 09.03.04 «Программная инженерия»

Тема: Серверная часть платформы для проведения онлайн-турниров по киберспорту

Студент: Хариаханов А. М.

Группа: ИКБО-03-22

Работа представлена к защите 05.12.24 (дата) / /
(подпись и ф.и.о. студента)

Руководитель: Синицын А.В.

Работа допущена к защите (дата) / /
(подпись и ф.и.о. рук-ля)

Оценка по итогам защиты:

 / /
 / /

(подписи, дата, ф.и.о., должность, звание, уч. степень двух преподавателей, принявших
защиту)



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ)

Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ЗАДАНИЕ

на выполнение курсовой работы

по дисциплине: Разработка серверных частей интернет-ресурсов
направления профессиональной подготовки: Программная инженерия (09.03.04)

Студент: Хариханов Анди Майрбекович

Группа: ИКБО-03-22

Срок представления к защите: 06.12.2024

Руководитель: Синицын Анатолий Васильевич, старший преподаватель

Тема: Серверная часть платформы для проведения онлайн-турниров по киберспорту

Исходные данные: используемые технологии: HTML5, CSS3, Django, Django REST framework, Pycharm, SQL СУБД, наличие: межстраничной навигации, внешнего вида страниц, соответствующего современным стандартам веб-разработки, использование паттерна проектирования (MVT). Нормативный документ: инструкция по организации и проведению курсового проектирования СМКО МИРЭА 7.5.1/04.И.05-18.

Перечень вопросов, подлежащих разработке, и обязательного графического материала:

1. Провести анализ предметной области разрабатываемого веб-приложения. 2. Обосновать выбор технологий разработки веб-приложения. 3. Разработать архитектуру веб-приложения на основе выбранного паттерна проектирования. 4. Реализовать слой серверной логики веб-приложения с применением выбранной технологии. 5. Реализовать слой логики базы данных. 6. Провести тестирование серверной части веб-приложения 7. Создать презентацию по выполненной курсовой работе.

Руководителем произведён инструктаж по технике безопасности, противопожарной технике и правилам внутреннего распорядка.

Зав. кафедрой ИиППО: [подпись] /Р. Г. Болбаков/, «16» сентября 2024 г.

Задание на КР выдал: [подпись] /А.В. Синицын/, «16» сентября 2024 г.

Задание на КР получил: [подпись] /А.М. Хариханов/, «16» сентября 2024 г.

Руководитель курсовой работы: старший преподаватель А.В. Сеницын

Хариханов А.М., Курсовая работа направления подготовки «Программная инженерия» на тему «**Серверная часть платформы для проведения онлайн-турниров по киберспорту**»: М. 2024 г., МИРЭА – Российский технологический университет (РТУ МИРЭА), Институт информационных технологий (ИИТ), кафедра инструментального и прикладного программного обеспечения (ИиППО) – 32 стр., 24 рис., 1 табл., 21 источн. 1 прил.

Ключевые слова: ИНТЕРНЕТ-РЕСУРС, ПАТТЕРН MVC, DJANGO, BACKEND РАЗРАБОТКА, КИБЕРСПОРТ, ОНЛАЙН-ТУРНИРЫ.

Целью работы является создание интернет ресурса на выбранную тему «Серверная часть платформы для проведения онлайн-турниров по киберспорту» для дальнейшего развития проекта

Kharikhanov A.M., Course work in the field of preparation “Software Engineering” on the topic “Server part of the platform for conducting online eSports tournaments”: M. 2023, MIREA - Russian Technological University (RTU MIREA), Institute of Information Technologies (IIT)), Department of Instrumental and Application Software (IiPPO) – 32 pages, 24 figures, 1 table, 21 source 1 adj.

Key words: INTERNET RESOURCE, MVC PATTERN, DJANGO, BACKEND DEVELOPMENT, ESPORTS, ONLINE TOURNAMENTS.

The goal of the work is to create an Internet resource on the selected topic “Server part of the platform for conducting online eSports tournaments” for the further development of the project

РТУ МИРЭА: 119454, Москва, пр-т Вернадского, д. 78

кафедра инструментального и прикладного программного обеспечения (ИиППО)

Тираж: 1 экз. (на правах рукописи)

Файл: «ПЗ_РСЧИР_ИКБО-03-22_Хариханов АМ.pdf», исполнитель

Хариханов А.М.

© А.М. Хариханов

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ	6
2. ВЫБОР И ОБОСНОВАНИЕ ТЕХНОЛОГИЙ	8
3. РАЗРАБОТКА АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ НА ОСНОВЕ ВЫБРАННОГО ПАТТЕРНА	11
4. РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ИНТЕРНЕТ РЕСУРСА	16
4.1 Разработка интернет ресурса	16
4.2 тестирование интернет ресурса	18
ЗАКЛЮЧЕНИЕ	30
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ	31
ПРИЛОЖЕНИЕ	33

ВВЕДЕНИЕ

Целью курсовой работы является создание серверной [1] части платформы для проведения онлайн-турниров по киберспорту, что позволит обеспечить стабильную и эффективную работу турниров. Актуальность исследования обусловлена стремительным ростом интереса к киберспорту [2], а также необходимостью создания надёжных инструментов для управления турнирами.

В условиях высокой конкуренции между игровыми платформами и организаторами турниров наличие качественной серверной инфраструктуры становится ключевым фактором успеха. Объектом исследования является процесс организации онлайн-турниров по киберспорту, а предметом - серверная часть платформы, обеспечивающая функциональность и взаимодействие всех участников турнира. Предполагаемый результат работы включает в себя создание серверной части, которая будет включать в себя функционал для регистрации участников, управления турнирами между различными киберспортивными командами, а так же создание новостной ленты мира киберспорта.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Для того что бы провести анализ предметной области, необходимо выделить основной функционал, который существует в похожих программных решений на тему создание серверной части платформы для проведения онлайн-турниров по киберспорту.

CYBERMOS: [3] Cybermos, представляет собой официальную платформу для проведения турниров в Москве. По большей части оно является информационным ресурсом где можно узнать различные новости из мира киберспорта Москвы. Поучаствовать в турнирах довольно сложно, так как приходится переходить на сторонние ресурсы сайта и искать информацию там

turnir-online.ru: [4] данный сайт скорее чистой серверной частью. Данный ресурс представляет собой огромным списком названий турниров (не всегда киберспортивных) где любой человек может вручную записать себя в турнир. Сайт имеет огромную информационную загруженность, но низкий уровень реализации. Многие функции сайта очень сложны в понимании и требуют дополнительного изучения, что плохо сказывается на конкурентно способности

cybersport.ru: [5] один из крупнейших российских медиа-порталов о киберспорте, на сайте можно узнать информацию о всех последних крупных турниров. Главным недостатком является то, что оглашается информация только о тех турнирах которые будут проведены, однако поучаствовать в турнире нельзя. Таким образом данный интернет ресурс хоть и является довольно крупным интернет порталом, но не представляет возможность участвовать или создать собственные турниры

Проанализируем конкурентов и составим на основе их плюсов и минусов матрицу смежности

Таблица 1 - таблица смежности

Название конкурента	Платформа для проведения киберспортивных турниров	Новостная лента	Возможность почувствовать в турнире	Возможность создать турнир
Cybermos	+	+	-+	-
turnir-online	+	-	+	+
cybersport	+	+	-	-

Можем заметить что сайт turnir-online кажется наилучшим конкурентом, но это достигается лишь за счет огромного функционала сайта в формате матрицы смежности. На практике он нуждается в огромной доработке.

Проведя анализ конкурентов можно выделить множество преимуществ, по сравнению с другими аналогами, а так же их минусы. Для создания актуального и конкурентноспособного интернет ресурса мы возьмём все преимущества которые были найдены и избавимся от недостатков

2. ВЫБОР И ОБОСНОВАНИЕ ТЕХНОЛОГИЙ

В данной главе будет проведён анализ выбранных технологий и паттернов проектирования, которые будут использоваться для разработки серверной части платформы для онлайн-турниров по киберспорту. Обоснование выбора каждой технологии будет основано на их функциональности, совместимости и преимуществах в контексте разрабатываемого программного продукта.

HTML5

HTML5 [6] предоставляет новые семантические элементы, улучшенную поддержку мультимедиа и возможности для создания адаптивных интерфейсов. Это особенно важно для платформы, где пользователи взаимодействуют через браузер.

Django REST Framework

Django REST Framework [7] является мощным инструментом для создания веб-API на основе Django. Он предоставляет множество возможностей, которые упрощают разработку RESTful API, что особенно важно для приложений, взаимодействующих с клиентами через интернет.

Django

Django [8]— это высокоуровневый веб-фреймворк на Python, который позволяет быстро разрабатывать безопасные и масштабируемые веб-приложения.

Преимущества Django:

- Быстрая разработка: Django следует принципу "batteries included", что означает наличие множества встроенных функций, таких как аутентификация пользователей, админ-панель и ORM (Object-Relational Mapping).
- Безопасность: Фреймворк предлагает защиту от распространенных уязвимостей, таких как SQL-инъекции и XSS [9].
- Масштабируемость: Django позволяет легко масштабировать приложение по мере роста числа пользователей.

PyCharm

PyCharm [10] — это интегрированная среда разработки (IDE) для Python, которая предоставляет мощные инструменты для работы с Django.

Преимущества PyCharm:

- Поддержка Django: Встроенные инструменты для работы с Django упрощают создание проектов и управление ими.
- Интеллектуальная подсказка кода: Упрощает процесс написания кода благодаря автозаполнению и подсветке синтаксиса.
- Инструменты отладки: Позволяют эффективно находить и устранять ошибки в коде.

SQLite СУБД

Для хранения данных о турнирах, игроках и матчах будет использоваться реляционная база данных на основе SQL [11].

Преимущества SQL СУБД:

- Структурированность данных: Позволяет организовать данные в таблицы с четкими связями между ними.
- Запросы на языке SQL: Обеспечивает мощные возможности для извлечения и манипуляции данными.

Паттерн проектирования MVT (Model-View-Template)

В рамках разработки на Django используется паттерн проектирования MVT (Model-View-Template) [12], который является адаптацией классического MVC [13].

- Model (Модель): Определяет структуру данных приложения. В Django модели представляют собой классы, которые описывают таблицы базы данных. Они обеспечивают взаимодействие с базой данных через ORM.
- View (Представление): Обрабатывает запросы от пользователей и возвращает соответствующие ответы. В Django представления определяют логику обработки запросов и формируют ответ в виде HTML или других форматов.

- Template (Шаблон): Отвечает за отображение данных пользователю. Шаблоны позволяют разделить логику приложения от его представления, что упрощает разработку интерфейса.

Выбор технологий обоснован их функциональностью и возможностями масштабирования. Использование паттерна проектирования MVT обеспечивает четкое разделение ответственности между компонентами приложения, что упрощает разработку и тестирование. Эти технологии в сочетании создадут надежную платформу для проведения онлайн-турниров по киберспорту, способную удовлетворить потребности пользователей.

3. РАЗРАБОТКА АРХИТЕКТУРЫ ПРИЛОЖЕНИЯ НА ОСНОВЕ ВЫБРАННОГО ПАТТЕРНА

Для разработки архитектуры веб-приложения следует выявить бизнес-правила, на основе которых будет строиться информационная система. На рисунке 1 представлена диаграмма вариантов использования UML [14] для проектируемой информационной системы:

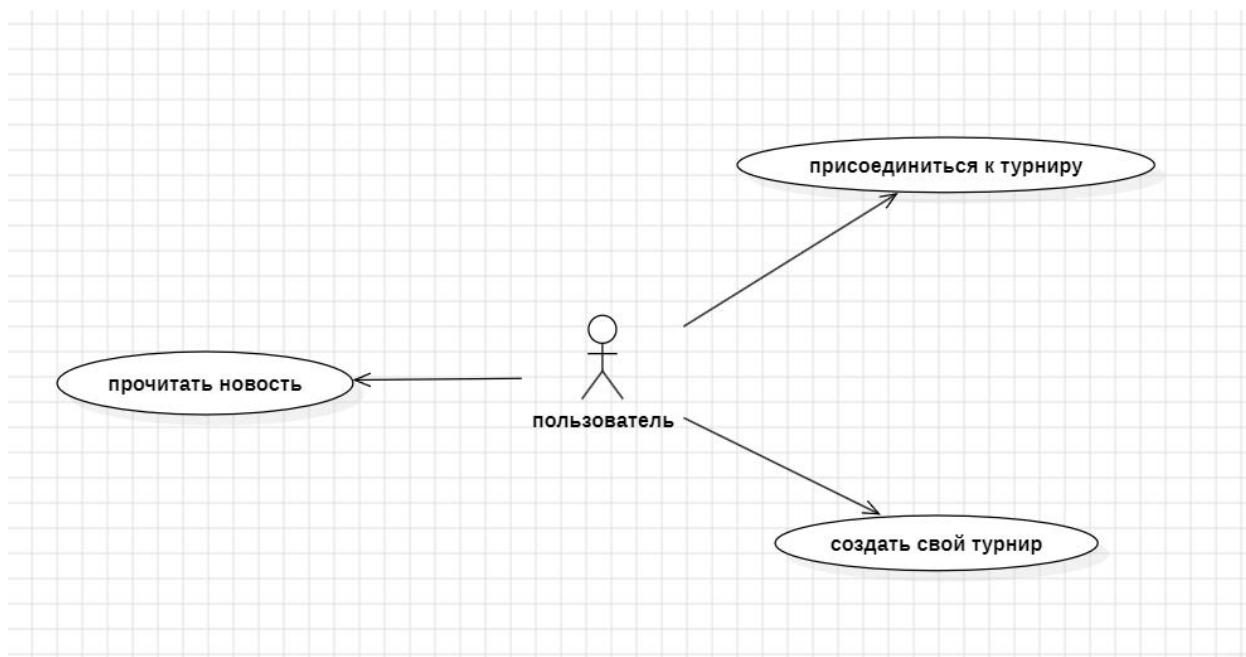


Рисунок 1 – Диаграмма вариантов использования веб-приложения

Для разработки веб-приложения в предыдущей главе был выбран паттерн MVT

MVT включает три ключевых компонента: Модель, Представление и Шаблон [15].

Модель отвечает за структуру данных и бизнес-логику приложения. В Django модели реализуются как Python-классы, которые определяют поля и поведение данных. Представление обрабатывает запросы пользователя и формирует ответы. В Django представления реализуются как функции или классы, которые принимают HTTP-запросы и возвращают HTTP-ответы. Шаблоны отвечают за визуализацию данных. Применяя данный архитектурный паттерн, создадим 3 приложения для отображения различных элементов веб сайта.

Первое приложение является главной страницей связывающей основные компоненты интернет ресурса между собой в виде межстраничной навигации. Так как на главной странице содержится только навигационная панель, то для корректной работы достаточно использовать элемент Template

Второе приложение представляет собой список турниров. Для реализации данного приложения потребовалось создать несколько классов. Рассмотрим созданные сущности [16] на рисунке 2

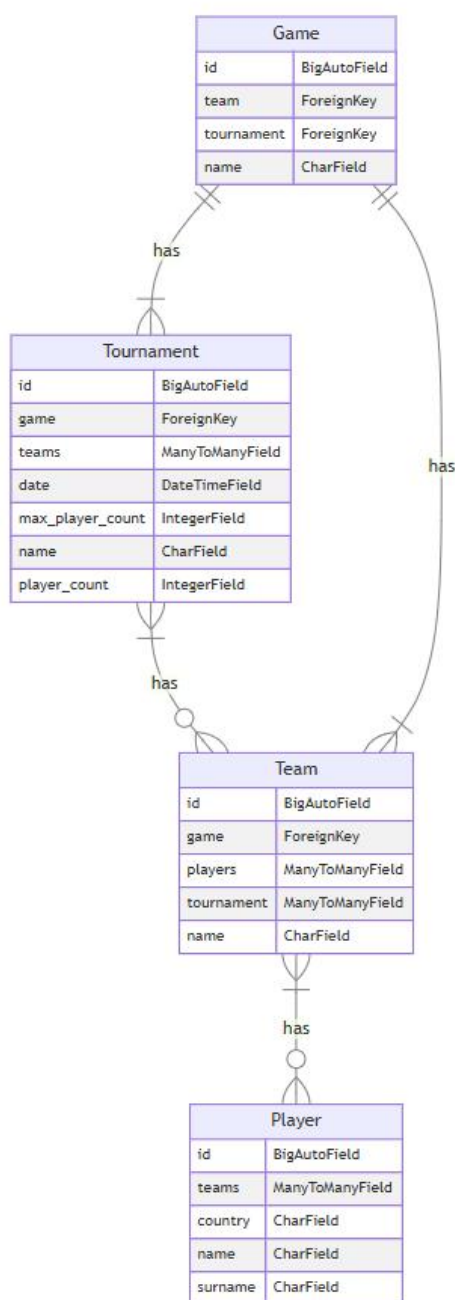


Рисунок 2 – Диаграмма сущностей приложения списка турниров

Внимательно рассмотрим представленную диаграмму. Начнем с центрального элемента — таблицы турниров. Эта таблица включает в себя две другие таблицы: таблицы игр и команд. Таблица турниров содержит такие поля, как идентификатор (id), время начала, название турнира, а также максимальное и текущее количество игроков. Эти данные необходимы для организации турниров.

Таблица игр представляет собой список игр, которые используются в рамках турниров, в то время как таблица команд включает информацию о игроках, названиях команд и турнирах, в которых они участвуют. Таблица игроков содержит сведения о участниках турниров через команды.

Все эти элементы составляют Model в парадигме MVT. Элемент View представляет собой набор функций, необходимых для работы с элементом Model, а элемент Template позволяет выводить информацию на экран. Более подробно о элементах View и Template можно узнать в соответствующем разделе приложения

Рассмотрим 3 приложение. 3 приложение представляет собой ленту новостей. Изучим диаграмму сущностей приложения новостей на рисунке 3

NewsArticle	
id	BigAutoField
content	TextField
published_date	DateTimeField
title	CharField

Рисунок 3 – Диаграмма сущности приложения ленты новостей

В третьем приложении элемент model представляет собой единую таблицу, в которой хранится информация о новостях. Элементы views

используются для корректного отображения этих новостей, а `templates` предназначены для вывода информации на экран. Более подробно о элементах `views` и `templates` можно узнать в соответствующем разделе приложения

Теперь построим диаграммы классов [17] которые являются элементом `views` в паттерне MVT. Пропустим приложение 1, так как в нем нет элемента `views` и сразу перейдём к приложению 2 (Рис 4)

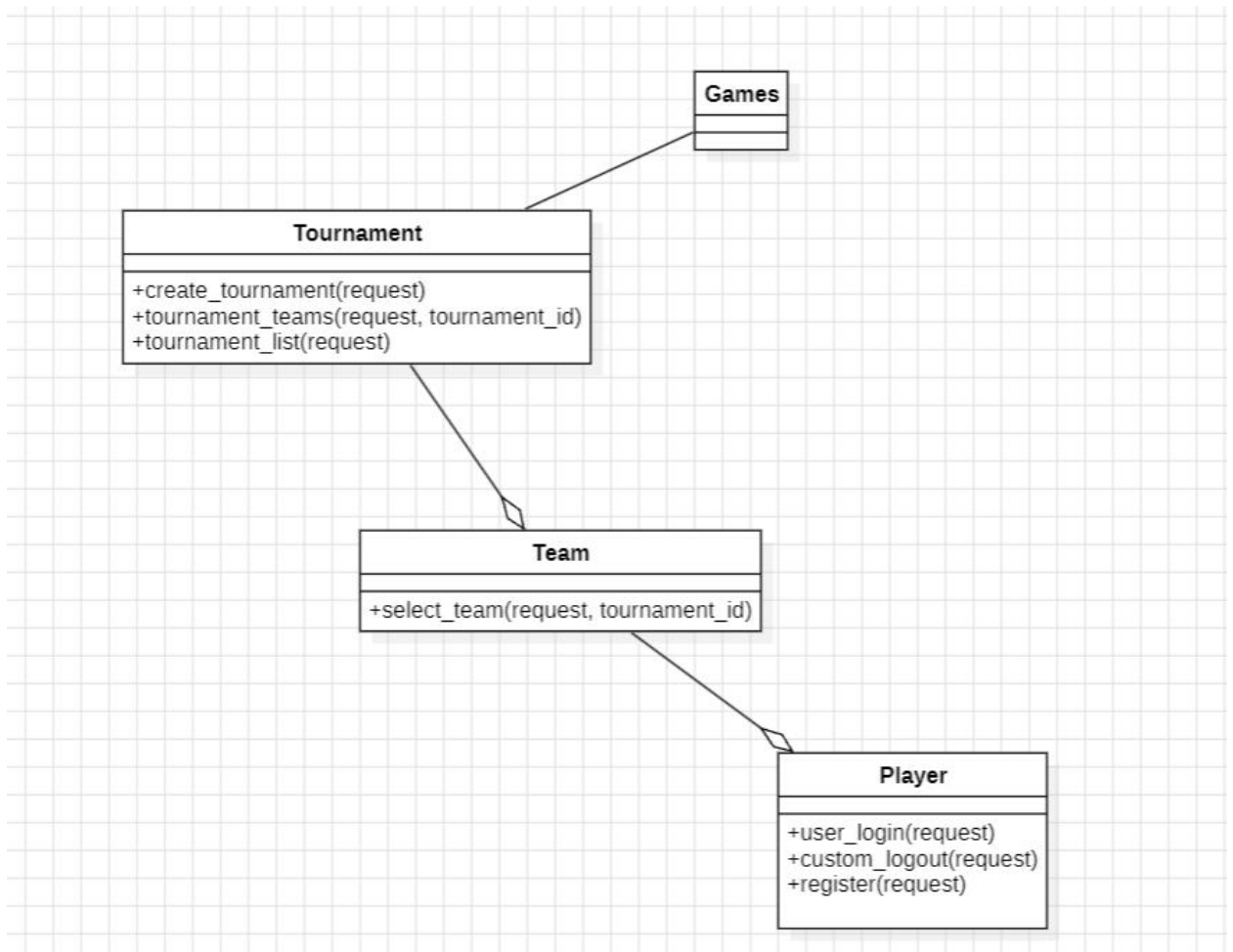


Рисунок 4 - диаграмма классов страницы списка турниров

На данной диаграмме мы можем наблюдать, как турниры включают в себя команды, а команды, в свою очередь, состоят из игроков. Класс игр представлен как дополнительный элемент в этой структуре. Кроме того, на диаграмме отображены используемые методы классов. Более подробно изучить указанные методы можно в соответствующем разделе приложения

Теперь перейдём к приложению 3, где находится всего 1 класс (Рис. 5)



Рисунок 4 - диаграмма классов страницы ленты новостей

Как видно, данная диаграмма классов по размерам не отличается от диаграммы сущностей на той же странице. В ней используются всего два метода, которые позволяют выводить новости в различных форматах. Изучить код этих методов можно в соответствующем разделе приложения

Таким образом данные диаграммы представляют собой реализацию паттерна MVT необходимого для полноценной разработки архитектуры серверной части интернет ресурса.

4. РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ИНТЕРНЕТ РЕСУРСА

Теперь рассмотрим разработку серверной части интернет-ресурса для проведения киберспортивных турниров. В предыдущих главах мы уже определили список технологий, которые будем использовать, а также разработали необходимую архитектуру на основе паттерна MVT. Весь код можно будет найти в приложениях в конце документа.

4.1 Разработка интернет ресурса

Для разработки интернет ресурса потребовалось создать 3 различных приложения (помимо корневого) для полноценной разработки

Приложение main - представляет собой главную страницу с возможностью перехода в другие приложения

Приложение news - представляет собой новостную ленту с полноценной реализацией паттерна MVT

Приложение tournaments - представляет собой ключевое приложение интернет ресурса где происходит регистрация на турниры и их создание с полной реализацией паттерна MVT

Для разработки серверной части интернет платформы необходимо создать базу данных для сохранения различной информации необходимой для работы интернет ресурса. Для создания базы данных была использована sqlite, так как она уже встроена в фреймворк Django и проста в своем использовании.

Рассмотрим код файлов Models.py данных в листинге 1

Листинг 1 – models.py

```
class Game(models.Model):
    name = models.CharField(max_length=100)

class Player(models.Model):
    name = models.CharField(max_length=100)
    surname = models.CharField(max_length=100)
    country = models.CharField(max_length=100)
    game = models.ForeignKey(Game, on_delete=models.CASCADE)
    tournaments = models.ManyToManyField('Tournament',
related_name='players')
```


Продолжение листинга 1

```
class Team(models.Model):
    name = models.CharField(max_length=100)
    players = models.ManyToManyField(Player,
related_name='teams', blank=True)
    game = models.ForeignKey(Game, on_delete=models.CASCADE,
null=True, blank=True)

class Tournament(models.Model):
    name = models.CharField(max_length=100)
    game = models.ForeignKey(Game, on_delete=models.CASCADE)
    teams = models.ManyToManyField(Team)
    player_count = models.IntegerField(default=0)
    max_player_count = models.IntegerField(default=8)
    date = models.DateTimeField()

class NewsArticle(models.Model):
    title = models.CharField(max_length=200)
    content = models.TextField()
    published_date = models.DateTimeField(auto_now_add=True)
```

Как мы можем видеть она полностью совпадает с диаграммами построенными в прошлой главе.

Для работы с базой данных и прочих функций сайта были созданы различные методы в файлах `view.py` каждого приложения с полноценной реализацией паттерна MVT.

Рассмотрим часть кода элементов View в листинге 2. так как размер всего кода слишком велик что бы вставлять его в основную часть, то мы рассмотрим лишь 1 метод

Листинг 2 – Код элементов View

```
def select_team(request, tournament_id):
    try:
        tournament = Tournament.objects.get(id=tournament_id)
    #
    except ObjectDoesNotExist:
        return redirect('tournament_list')

    teams = tournament.teams.all()

    if request.method == 'POST':
        selected_team_id = request.POST.get('team')
        selected_team = Team.objects.get(id=selected_team_id)
```

Продолжение листинга 2

```
player, created = Player.objects.get_or_create(
    name=request.user.username,
    surname='',
    country='',
    game=tournament.game,
)

current_team = None
for team in tournament.teams.all():
    if player in team.players.all():
        current_team = team
        break

if current_team:
    current_team.players.remove(player)

selected_team.players.add(player)

unique_players = set()
for team in tournament.teams.all():
    unique_players.update(team.players.all())

tournament.player_count = len(unique_players)
tournament.save()

return redirect('tournament_list')

return render(request, 'tournaments/select_team.html',
{'tournament': tournament, 'teams': teams})
```

Данный метод обрабатывает выбор пользователем команды для турниров. Для полноценного изучения кода можно перейти к приложению 10

4.2 тестирование интернет ресурса

Начнем тестирование с точки зрения пользователя, который только что зашел на интернет-ресурс. Само тестирование[18] будет проходить вручную методом чёрного ящика [19] это означает что мы будем тестировать приложение так, как будто не знаем кода приложения.

При входе на интернет ресурс нас встречает главная страница. Проверим что главная страница в основном представляет собой навигационную панель (Рис. 6)

Главная страница

Перейти к турнирам: [Турниры](#)

Перейти к новостям: [Новости](#)

Рисунок 6 - главная страница

Как мы видим главная страница, как было описано ранее, представляет собой навигационную панель из которой можно попасть на 2 основные страницы, страницу турниров и новостей.

Проверим работу разработанной страницы турниров, для этого перейдём по ссылке на страницу турниров (рис. 7)

Список турниров

Вход в аккаунт

Username:

Password:

[Зарегистрироваться](#)
[обратно](#)

- IEM Rio - Игра: CS2 - Количество игроков: 0 / 8 - Дата: 24.12.2024 18:00 - [Посмотреть команды](#)

Рисунок 7 - Страница Списка турниров

Мы попали на страницу списка турниров. Внизу страницы можем увидеть заранее созданный турнир. Так же можем увидеть возможность зарегистрироваться и войти в аккаунт. Сперва посмотрим какие команды участвуют в турнире (Рис. 8)

Команды турнира: IEM Rio

- **NAVI**
 - Нет игроков в команде.
- **VIRTUS.PRO**
 - Нет игроков в команде.

[Назад к списку турниров](#)

Рисунок 8 - страница команд турнира

Как было видно из страницы списка турниров, количество игроков равно 0 из 8 и мы действительно можем увидеть 2 команды, в каждой из которой имеется 0 игроков. Обе команды являются таблицами команд которые были описаны ранее. Теперь перейдём обратно и нажмём кнопку регистрации что бы создать свой аккаунт (Рис. 9)

Создать аккаунт

Username: Required. 150 characters or fewer. Letters, digits and @/./+/_ only.

Email:

Password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation: Enter the same password as before, for verification.

[Назад к списку турниров](#)

Рисунок 9 - страница регистрации

Придумаем имя пользователя, вводим свою электронную почту, придумываем пароль и подтверждаем его. Наконец нажимаем кнопку регистрации и нас автоматически переносит на страницу списка турниров (Рис. 8). Когда мы создали нового пользователя он сразу же записался в таблицу игроков созданной базы данных. Заметим что все шаги сопровождаются возможностью вернуться обратно благодаря межстраничной навигации

Список турниров

Добро пожаловать, BestGamer!

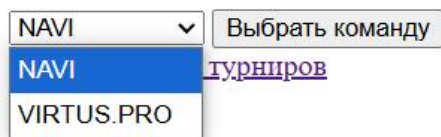
[Выйти](#)
[обратно](#)
[Создать собственный турнир](#)

- IEM Rio - Игра: CS2 - Количество игроков: 0 / 8 - Дата: 24.12.2024 18:00 - [Выбрать команду](#) [Посмотреть команды](#)

Рисунок 10 - страница списка турниров после входа в аккаунт.

Заметим, что страница преобразилась. Страница теперь приветствует пользователя. Больше нет кнопки входа и регистрации, а вместо неё теперь есть кнопка выхода из аккаунта. Так же появилась ссылка для создания собственного турнира, к ней мы вернёмся по позже. Но нас больше всего интересует кнопка выбора команды. Нажмём на неё (Рис. 9)

Выберите команду для турнира IEM Rio



NAVI ▾ [Выбрать команду](#)

NAVI [турниров](#)

VIRTUS.PRO

Рисунок 10 - выбор команд

Мы видим страницы выбора команд с выпадающим списком. Выберем 1 команду и вернёмся обратно. (рис 12).

Список турниров

Добро пожаловать, BestGamer!

[Выйти](#)

[обратно](#)

[Создать собственный турнир](#)

- IEM Rio - Игра: CS2 - Количество игроков: 1 / 8 - Дата: 24.12.2024 18:00 - [Выбрать команду](#) [Посмотреть команды](#)

Рисунок 12 - список турниров

На этот раз увидим что количество игроков стало 1 из 8 это говорит о том что методы обновления количества игроков разработанные для этой страницы работают корректно. Нажмём на ссылку просмотра команд что бы убедиться что бы зарегистрировались на турнир (рис 13).

Команды турнира: IEM Rio

- **NAVI**
 - BestGamer
- **VIRTUS.PRO**
 - Нет игроков в команде.

[Назад к списку турниров](#)

Рисунок 13 - просмотр команд к турниру.

Можем заметить что мы действительно появились в одной из команд к турниру. Теперь создадим несколько других аккаунтов и зайдём в различные команды (Рис. 14).

Команды турнира: IEM Rio

- **NAVI**
 - BestGamer
 - user3
- **VIRTUS.PRO**
 - user1
 - user2

[Назад к списку турниров](#)

Рисунок 14 - команды турнира

Заметим что на турнир зашло еще 3 человека. Вернёмся на главную страницу турниров и проверим что количество игроков так же увеличилось (Рис. 15).

Список турниров

Добро пожаловать, BestGamer!

Выйти

[обратно](#)

[Создать собственный турнир](#)

- IEM Rio - Игра: CS2 - Количество игроков: 4 / 8

Рисунок 15 список турниров

Заметим что количество игроков в турнире соответствует количеству человек в команде.


Теперь создадим собственный турнир, для этого нажмём на ссылку создания турнира, все еще находясь в своем аккаунте (Рис. 16).

Создать новый турнир

Название турнира:

Название игры:

Максимальное количество игроков:

Дата турнира: 

[Назад к списку турниров](#)

Рисунок 16 - создание собственного турнира

Нас встречает страница создания турниров. После введения названия турнира, названия игры, максимального количества игроков и даты можем теперь нажать на кнопку создания турнира, после чего вернемся обратно что бы проверить что все действительно работает (Рис. 17)

Список турниров

Добро пожаловать, BestGamer!

[обратно](#)

[Создать собственный турнир](#)

- IEM Rio - Игра: CS2 - Количество игроков: 4 / 8 - Дата: 24.12.2024 18:00 - [Выбрать команду](#) [Посмотреть команды](#)
- Турнир для меня и моих друзей - Игра: Roblox - Количество игроков: 0 / 4 - Дата: 21.12.2024 18:03 - [Выбрать команду](#) [Посмотреть команды](#)

Рисунок 17 - проверка создания турнира

Сравнив имена турнира с лёгкостью находим только что созданный нами турнир, который теперь находится в самом низу списка. Как в прошлый раз теперь выберем команду и посмотрим список игроков (Рис. 18)

Выберите команду для турнира Турнир для меня и моих друзей

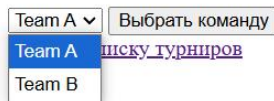


Рисунок 18 - присоединение к команде

Заметим что команды сами создались, так как мы не являемся капитанами команд, то вместо них появились команды Team A и Team B

В последний раз проверим что мы действительно зашли в команду. Для этого вновь вернёмся назад и нажмём на ссылку просмотра команд у созданного нами турнира, но перед этим еще раз убедимся в том что количество игроков отображается корректно (Рис. 19)

Список турниров

Добро пожаловать, BestGamer!

Выйти

[обратно](#)

[Создать собственный турнир](#)

- ИЕМ Rio - Игра: CS2 - Количество игроков: 4 / 8 - Дата: 24.12.2024 18:00 - [Е](#)
- Турнир для меня и моих друзей - Игра: Roblox - Количество игроков: 1 / 4 -

Рисунок 19 - список турниров

Все правильно, теперь посмотрим кто находится в командах (Рис. 20)

Команды турнира: Турнир для меня и моих друзей

- **Team A**
 - BestGamer
- **Team B**
 - Нет игроков в команде.

[Назад к списку турниров](#)

Рисунок 20 - участники команд

Все работает корректно, таким образом можно сделать вывод, что созданный нами турнир функционирует аналогично созданному заранее. Теперь выйдем из аккаунта и посмотрим на то, как выглядит страница списка турниров (Рис. 21)

Список турниров

Вход в аккаунт

Username:

Password:

[Зарегистрироваться](#)
[обратно](#)

- ИЕМ Rio - Игра: CS2 - Количество игроков: 4 / 8 - Дата: 24.12.2024 18:00 - [Посмотреть команды](#)
- Турнир для меня и моих друзей - Игра: Roblox - Количество игроков: 1 / 4 - Дата: 21.12.2024 18:03 - [Посмотреть команды](#)

Рисунок 21 - Список турниров без авторизации

Как можно увидеть, страница пришла в изначальное положение как когда мы в первый раз зашли на страницу

Теперь изучим страницу новостей, для этого вернёмся на главную страницу и перейдём по ссылке на страницу ленты новостей (Рис 22).

Новости киберспорта

[обратно](#)

- **Киберспортивный клуб РТУ МИРЭА**

Киберспортивный клуб РТУ МИРЭА, основанный в 2012 ...

Dec. 1, 2024, 4:31 p.m.

- **Финал чемпионата Москвы по CS2:**

23 ноября 2024 года в рамках фестиваля разработчик...

Dec. 1, 2024, 4:31 p.m.

- **РТУ МИРЭА стал чемпионом Всероссийской киберспортивной студенческой лиги 2024**

16 июня 2024 года РТУ МИРЭА одержал победу на фина...

Dec. 1, 2024, 4:31 p.m.

Рисунок 22 - Лента новостей

При входе на страницу ленты новостей нас встречают несколько новостей. По прошлой главе мы помним что новости хранятся в базе данных интернет ресурсы и состоят из 1 таблицы. Мы можем увидеть только первые 50 символов основного текста новости. Давайте перейдём по ссылке на одну из новостей что бы прочитать её полностью (Рис 23).

Киберспортивный клуб РТУ МИРЭА

Киберспортивный клуб РТУ МИРЭА, основанный в 2012 году, продолжает развивать свои достижения. С момента своего создания клуб прошел путь от небольших локальных турниров до участия в крупных межвузовских соревнованиях. В 2022 году клуб был удостоен звания «Киберспортивный клуб года» на Российской национальной премии «Студент года». Современная инфраструктура клуба, включая Киберзону с тремя игровыми площадками, способствует подготовке студентов к успешным выступлениям в различных дисциплинах

Опубликовано: Dec. 1, 2024, 4:31 p.m. [Назад](#)

Рисунок 23 - подробности новости

Мы оказались на странице подробности новостей. Сюда передаётся id новости после чего вся новость загружается из БД и передаётся на экран.

Наконец воспользуемся Django REST framework что бы посмотреть на созданные аккаунты (Рис 24)

Django REST framework

```
},
{
    "url": "http://127.0.0.1:8000/users/7/",
    "username": "BestGamer",
    "email": "BestGamer@mail.ru",
    "is_staff": false
},
{
    "url": "http://127.0.0.1:8000/users/8/",
    "username": "user1",
    "email": "7@mail.ru",
    "is_staff": false
},
{
    "url": "http://127.0.0.1:8000/users/9/",
    "username": "user2",
    "email": "7@mail.ru",
    "is_staff": false
},
{
    "url": "http://127.0.0.1:8000/users/10/",
    "username": "user3",
    "email": "7@mail.ru",
    "is_staff": false
}
]
```

Рисунок 24 - использование Django REST framework

Таким образом мы прошлись по всему функционалу вэб-ресурса глазами пользователя. Каждый наш шаг описывал различные элементы которые были разработаны и сразу же протестированы. Тестирование показало что весь функционал работает корректно

ЗАКЛЮЧЕНИЕ

В ходе курсовой работы было выполнено множество задач. Мы начали с анализа предметной области, в рамках которого изучили конкурентов, выявили их сильные стороны, которые были учтены в нашем проекте, а также нивелировали их слабости.

Затем мы выбрали и обосновали ряд технологий, которые помогли нам достичь актуальности и конкурентоспособности. После этого была разработана архитектура проекта на основе паттерна MVT, что позволило нам четко структурировать код. В этом нам помогли различные диаграммы, наглядно демонстрирующие архитектуру, которую мы использовали в разработке серверной части интернет-ресурса.

Итогом работы стала разработка серверной части платформы для проведения киберспортивных турниров, необходимой для полноценного создания интернет-ресурса, а так же написан отчет по ГОСТу [20]. Код был загружен на github [21]

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. sim-networks.com [электронный ресурс] URL: <https://www.sim-networks.com/ru/blog/it-infrastructure> (дата обращения 15.11.2024).
2. киберспорт.рф [электронный ресурс] URL: <https://xn--90aihxfgcgn.xn--p1ai/esport/> (дата обращения 11.11.2024).
3. Cybermos [электронный ресурс] URL: <https://cybermos.ru/> (дата обращения 10.11.2024).
4. turnir-online [электронный ресурс] URL: <http://www.turnir-online.ru/> (дата обращения 10.11.2024).
5. cybersport [электронный ресурс] URL: <https://www.cybersport.ru/> (дата обращения 12.11.2024).
6. html5book [электронный ресурс] URL: <https://html5book.ru/html5/> (дата обращения 20.11.2024).
7. Habr [электронный ресурс] URL: https://habr.com/ru/companies/yandex_praktikum/articles/561696/ (дата обращения 13.11.2024).
8. Djangoprojec [электронный ресурс] URL: <https://www.djangoproject.com/> (дата обращения 13.11.2024).
9. MDN web docs [электронный ресурс] URL: <https://developer.mozilla.org/ru/docs/Learn/Server-side/Django> (дата обращения 13.11.2024).
10. Habr [электронный ресурс] URL: <https://habr.com/ru/companies/vk/articles/823740/> (дата обращения 21.11.2024).
11. Habr [электронный ресурс] URL: <https://habr.com/ru/articles/720480/> (дата обращения 19.11.2024).
12. metanit.com [электронный ресурс] URL: <https://metanit.com/sql/mysql/> (дата обращения 1.11.2024).
13. Habr [электронный ресурс] URL: <https://habr.com/ru/articles/181772/> (дата обращения 19.11.2024).

14. Pythonpip [электронный ресурс] URL: <https://pythonpip.ru/django/django-mvt> (дата обращения 12.11.2024).
15. Flexberry [электронный ресурс] URL: https://flexberry.github.io/ru/fd_use-case-diagram.html (дата обращения 10.11.2024).
16. Habr [электронный ресурс] URL: <https://www.lucidchart.com/pages/ru/erd-diagram#> (дата обращения 21.11.2024).
17. cybersport [электронный ресурс] URL: <https://blog.skillfactory.ru/nakakih-yazykah-programmirovaniya-pishut-backend/#> (дата обращения 14.11.2024).
18. Habr [электронный ресурс] URL: https://habr.com/ru/companies/habr_career/articles/517812/ (дата обращения 15.11.2024).
19. Careerist [электронный ресурс] URL: <https://www.careerist.com/ru-insights/belyy-seryy-i-chernyy-yashchik> (дата обращения 15.11.2024).
20. ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу.
21. GITHUB [электронный ресурс] URL: <https://github.com/AndiXAM/backend> (дата обращения 13.11.2024).

Листинг А.1 - код urls.py

```
from django.contrib import admin
from django.urls import path, include
from django.contrib.auth.models import User
from rest_framework import routers, serializers, viewsets

class UserSerializer(serializers.HyperlinkedModelSerializer):
    class Meta:
        model = User
        fields = ['url', 'username', 'email', 'is_staff']

class UserViewSet(viewsets.ModelViewSet):
    queryset = User.objects.all()
    serializer_class = UserSerializer

URL conf.
router = routers.DefaultRouter()
router.register(r'users', UserViewSet)

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('main.urls')),
    #path('', include(router.urls)),
    path('api/', include('testpril.urls')),
    path('news/', include('news.urls')),
    path('api-auth/', include('rest_framework.urls',
namespace='rest_framework')),
]
```

Листинг А.2 - main Views.py

```
def index(request):
    return render (request, 'main/index.html')
```

Листинг А.3 - main index.html

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
```

```

        content="width=device-width, user-scalable=no,
initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
        <meta http-equiv="X-UA-Compatible" content="ie=edge">
        <title>Главная страница</title>
</head>
<body>
<h1>Главная страница</h1>
<p>
    Перейти к турнирам:
    <a
href="http://127.0.0.1:8000/api/tournaments/">Турниры</a> <br>
    Перейти к новостям:
    <a href="http://127.0.0.1:8000/news/">Новости</a>
</p>
</body>
</html>

```

Листинг A.4 - код news views.py

```

class Game(models.Model):
    from django.shortcuts import render, get_object_or_404
    from .models import NewsArticle

    def news_list(request):
        articles = NewsArticle.objects.all().order_by('-
published_date')
        return render(request, 'news/news_list.html', {'articles':
articles})

    def news_detail(request, article_id):
        article = get_object_or_404(NewsArticle, id=article_id)
        return render(request, 'news/news_detail.html', {'article':
article})

```

Листинг A.5 - код news models.py

```

class Game(models.Model):
    from django.db import models

    class NewsArticle(models.Model):
        title = models.CharField(max_length=200)
        content = models.TextField()
        published_date = models.DateTimeField(auto_now_add=True)

        def __str__(self):
            return self.title

```

Листинг А.6 - код news list_news.html

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Новости киберспорта</title>
</head>
<body>
  <h1>Новости киберспорта</h1>
  <a href="http://127.0.0.1:8000/">обратно</a>
  <ul>
    {% for article in articles %}
      <li>
        <h2><a href="{% url 'news_detail'
article.id %}">{{ article.title }}</a></h2>
        <p>{{ article.content|slice:"50" }}{% if
article.content|length > 50 %}...{% endif %}</p>
        <small>{{ article.published_date }}</small>
      </li>
    {% endfor %}
  </ul>
</body>
</html>
```

Листинг А.7 - код news detail_news

```
class Game(models.Model):
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>{{ article.title }}</title>
</head>
<body>
  <h1>{{ article.title }}</h1>
  <p>{{ article.content }}</p>
  <small>Опубликовано: {{ article.published_date }}</small>

  <a href="{% url 'news_list' %}">Назад</a>
</body>
</html>
```

Листинг А.8 - код tournament urls.py

```

from django.urls import path
from .views import tournament_list, create_tournament,
register, select_team, tournament_teams, user_login,
custom_logout

urlpatterns = [
    path('tournaments/', tournament_list,
name='tournament_list'),
    path('tournaments/create/', create_tournament,
name='create_tournament'),
    path('register/', register, name='register'),
    path('tournaments/<int:tournament_id>/select_team/',
select_team, name='select_team'),
    path('tournaments/<int:tournament_id>/teams/',
tournament_teams, name='tournament_teams'),
    path('login/', user_login, name='login'),
    path('logout/', custom_logout, name='logout'),
]

```

Листинг А.9 - код tournament models.py

```

from django.db import models
from django.core.exceptions import ValidationError
from django.utils import timezone

from django.db import models

class Game(models.Model):
    name = models.CharField(max_length=100)

    def __str__(self):
        return self.name

class Player(models.Model):
    name = models.CharField(max_length=100)
    surname = models.CharField(max_length=100)
    country = models.CharField(max_length=100)
    game = models.ForeignKey(Game, on_delete=models.CASCADE)
    tournaments = models.ManyToManyField('Tournament',
related_name='players')

    def __str__(self):
        return f"{self.name} {self.surname}"

class Team(models.Model):
    name = models.CharField(max_length=100)
    players = models.ManyToManyField(Player,
related_name='teams', blank=True)

```

```

        game = models.ForeignKey(Game, on_delete=models.CASCADE,
null=True, blank=True)

    def __str__(self):
        return self.name

class Tournament(models.Model):
    name = models.CharField(max_length=100)
    game = models.ForeignKey(Game, on_delete=models.CASCADE)
    teams = models.ManyToManyField(Team)
    player_count = models.IntegerField(default=0)
    max_player_count = models.IntegerField(default=8)
    date = models.DateTimeField()

    def clean(self):
        if self.player_count < 0 or self.player_count >
self.max_player_count:
            raise ValidationError('Player count must be
between 0 and max player count.')

    def __str__(self):
        return f"{self.name} - {self.game.name}"

```

Листинг А.10 - код tournament views.py

```

from django.shortcuts import render, redirect
from .models import Tournament, Game
from .forms import TournamentForm
from django.contrib.auth import login
from .forms import RegistrationForm
from .models import Tournament, Team, Player
from django.core.exceptions import ObjectDoesNotExist
from django.contrib.auth import login, authenticate
from .forms import CustomLoginForm
from django.contrib.auth.forms import AuthenticationForm
from django.contrib.auth import logout

def create_tournament(request):
    if request.method == 'POST':
        name = request.POST.get('name')
        game_name = request.POST.get('game_name')
        max_player_count =
int(request.POST.get('max_player_count'))
        date = request.POST.get('date')

```

```

        game, created =
Game.objects.get_or_create(name=game_name)

        tournament = Tournament(name=name, game=game,
player_count=0, max_player_count=max_player_count, date=date)
        tournament.save()

        team_a = Team.objects.create(name='Team A', game=game)
        team_b = Team.objects.create(name='Team B', game=game)

        tournament.teams.add(team_a, team_b)

        return redirect('tournament_list')

    return render(request, 'tournaments/create.html')

def tournament_list(request):
    tournaments = Tournament.objects.all()

    if request.method == 'POST':
        form = AuthenticationForm(data=request.POST)
        if form.is_valid():
            username = form.cleaned_data['username']
            password = form.cleaned_data['password']
            user = authenticate(request, username=username,
password=password)
            if user is not None:
                login(request, user)
                return redirect('tournament_list')
        else:
            form = AuthenticationForm()

    return render(request, 'tournaments/list.html',
{'tournaments': tournaments, 'form': form})

def register(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request, user)    user after registration
            return redirect('tournament_list')    # Redirect to
the tournament list page
        else:
            form = RegistrationForm()
    return render(request, 'register/register.html', {'form':
form})

def select_team(request, tournament_id):

```

```

try:
    tournament = Tournament.objects.get(id=tournament_id)
except ObjectDoesNotExist:
    return redirect('tournament_list')

teams = tournament.teams.all()

if request.method == 'POST':
    selected_team_id = request.POST.get('team')
    selected_team = Team.objects.get(id=selected_team_id)

    player, created = Player.objects.get_or_create(
        name=request.user.username,
        surname='',
        country='',
        game=tournament.game,
    )

    current_team = None
    for team in tournament.teams.all():
        if player in team.players.all():
            current_team = team
            break

    if current_team:
        current_team.players.remove(player)

    selected_team.players.add(player)

    unique_players = set()
    for team in tournament.teams.all():
        unique_players.update(team.players.all())

    tournament.player_count = len(unique_players)
    tournament.save()

    return redirect('tournament_list')

    return render(request, 'tournaments/select_team.html',
{'tournament': tournament, 'teams': teams})

def tournament_teams(request, tournament_id):
    try:
        tournament = Tournament.objects.get(id=tournament_id)
    except ObjectDoesNotExist:
        return redirect('tournament_list')

```

```

        teams = tournament.teams.all()

        return render(request, 'tournaments/team_list.html',
{'tournament': tournament, 'teams': teams})

def user_login(request):
    if request.method == 'POST':
        form = CustomLoginForm(request, data=request.POST)
        if form.is_valid():
            username = form.cleaned_data['username']
            password = form.cleaned_data['password']
            user = authenticate(request, username=username,
password=password)
            if user is not None:
                login(request, user)
                return redirect('tournament_list')
        else:
            form = CustomLoginForm()

    return render(request, 'login.html', {'form': form})

def custom_logout(request):
    if request.method == 'POST':
        logout(request) # Выход пользователя
        return redirect(request.META.get('HTTP_REFERER', '/'))

```

Листинг А.11 - код tournament create.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Создать турнир</title>
</head>
<body>
<h1>Создать новый турнир</h1>
<form method="post">
    {% csrf_token %}

    <label for="id_name">Название турнира:</label>
    <input type="text" name="name" id="id_name" required>
    <br>
    <label for="id_game">Название игры:</label>
    <input type="text" name="game_name" id="id_game" required>
    <br>
    <label for="id_max_player_count">Максимальное количество
игроков:</label>
    <input type="number" name="max_player_count"
id="id_max_player_count" min="1" max="100" value="8" required>
    <!-- Updated field -->
    <br>

```



```

        <label for="id_date">Дата турнира:</label>
        <input type="datetime-local" name="date" id="id_date"
required>
        <br>
        <button type="submit">Создать</button>
</form>
<a href="{% url 'tournament_list' %}">Назад к списку
турниров</a>
</body>
</html>

```

Листинг A.12 - код tournament list.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Список турниров</title>
</head>
<body>
<h1>Список турниров</h1>

{% if request.user.is_authenticated %}
    <p>Добро пожаловать, {{ request.user.username }}!</p>
    <form method="post" action="{% url 'logout' %}"> <!-- URL
для выхода -->
        {% csrf_token %}
        <button type="submit">Выйти</button>
    </form>
{% else %}
    <h2>Вход в аккаунт</h2>
    <form method="post">
        {% csrf_token %}
        {{ form.as_p }} <!-- Render the login form -->
        <button type="submit">Войти</button>
    </form>
    <a href="{% url 'register' %}">Зарегистрироваться</a>
<br><!-- Link to registration page -->
{% endif %}

<a href="http://127.0.0.1:8000/">обратно</a> <br>
{% if request.user.is_authenticated %}
<a href="{% url 'create_tournament' %}">Создать собственный
турнир</a>
{% endif %}

<ul>
    {% for tournament in tournaments %}
        <li>
            {{ tournament.name }} - Игра:
            {{ tournament.game.name }} -

```

```

        Количество игроков: {{ tournament.player_count }}
/ {{ tournament.max_player_count }} -
        Дата: {{ tournament.date|date:"d.m.Y H:i" }} -
        {% if request.user.is_authenticated %}
            <a href="{% url 'select_team'
tournament.id %}">Выбрать команду</a>
            {% endif %}
            <a href="{% url 'tournament_teams'
tournament.id %}">Посмотреть команды</a>
        </li>
        {% empty %}
            <li>Нет доступных турниров.</li>
        {% endfor %}
    </ul>

</body>
</html>

```

Листинг A.13 - код tournament login.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Вход в аккаунт</title>
</head>
<body>
<h1>Вход в аккаунт</h1>
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Войти</button>
</form>
<a href="{% url 'register' %}">Зарегистрироваться</a>
</body>
</html>

```

Листинг A.14 - код tournament select_team.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Выбор команды для {{ tournament.name }}</title>
</head>
<body>
<h1>Выберите команду для турнира {{ tournament.name }}</h1>
<form method="post">
    {% csrf_token %}
    <select name="team" required>
        {% for team in teams %}

```

```

        <option
value="{{ team.id }}">{{ team.name }}</option>
        {% empty %}
        <option disabled>Нет доступных команд</option>
        {% endfor %}
    </select>
    <button type="submit">Выбрать команду</button>
</form>
<a href="{% url 'tournament_list' %}">Назад к списку
турниров</a>
</body>
</html>

```

Листинг А.15 - код tournament select_team.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Команды турнира {{ tournament.name }}</title>
</head>
<body>
<h1>Команды турнира: {{ tournament.name }}</h1>

<ul>
    {% for team in teams %}
        <li>
            <strong>{{ team.name }}</strong>
            <ul>
                {% for player in team.players.all %}
                    <li>{{ player.name }}
{{ player.surname }}</li>
                    {% empty %}
                    <li>Нет игроков в команде.</li>
                {% endfor %}
            </ul>
        </li>
        {% empty %}
        <li>Нет команд в турнире.</li>
    {% endfor %}
</ul>

<a href="{% url 'tournament_list' %}">Назад к списку
турниров</a>
</body>
</html>

```