Министерство образования Калининградской области

Государственное автономное учреждение

Калининградской области

профессиональная образовательная организация

«Колледж предпринимательства»

**Курсовая работа**

**Тема: «Разработка лендинга "Видеоигра "DESTINY 2" "»**

Выполнил:

обучающийся гр. ИСП 20-22

специальность 09.02.07  
Информационные системы и программирование

Криворот А.С

руководитель

М.В. Зверев

Калининград

2023

# **СОДЕРЖАНИЕ**

3 см.

[СОДЕРЖАНИЕ 2](#_Toc17411)

[СПИСОК ТЕРМИНОВ И СОКРАЩЕНИЙ 3](#_Toc9519)

[ВВЕДЕНИЕ 5](#_Toc29924)

[1. ОПИСАНИЕ САЙТА 6](#_Toc29193)

[1.1 Шапка сайта 6](#_Toc9211)

[1.2 Баннер 10](#_Toc1913)

[1.3 Блок “ОСОБЕННОСТИ DESTINY 2” 12](#_Toc20658)

[1.4 Блок “ИСТОРИИ НА КАЖДОМ ШАГУ” 15](#_Toc15067)

[1.5 Блок “ОТЗЫВЫ” 18](#_Toc2063)

[1.6 Блок “Присоединяйся” 20](#_Toc9685)

[1.7 Подвал сайта 20](#_Toc783)

[ЗАКЛЮЧЕНИЕ 21](#_Toc13233)

# **СПИСОК ТЕРМИНОВ И СОКРАЩЕНИЙ**

|  |  |
| --- | --- |
| **Определение** | **Описание** |
| **1** | **2** |
| HTML | Язык гипертекстовой разметки |
| CSS | Каскадные таблицы стилей |
| JavaScript,JS | Язык программирования |
| anime.js | Библиотека для создания анимаций с использованием JavaScript |
| vanila - tilt | Библиотека для создания параллакс эффектов |
| fetch API | Интерфейс JavaScript для работы с запросами HTTP |
| PHP | Язык программирования |
| API | Набор способов и правил, по которым различные программы общаются между собой и обмениваются данными |
| waypoints | Библиотека для установки якорей на сайте к которым можно впоследствии обратится |
| JSON | текстовый формат обмена данными, основанный на JavaScript. |
| Сессия | механизм для сохранения информации на разных веб-страницах |
| AJAX | технология обращения к серверу без перезагрузки страницы |
| fetch запрос | http-запрос через fetch API |
| SQL | язык структурированных запросов, использующийся для работы с базой данных |
| DOM | объектная модель документа |
| Grid | пересекающийся набор горизонтальных и вертикальных линий, образующих колонки и строки |
| flexbox | технология для создания сложных гибких макетов за счёт правильного размещения элементов на странице. |
| Bungie.API | API Студии BUNGIE созданный для взаимодействия с данными DETINY 2 за её пределами и находящийся в открытом доступе. |
| **1** | **2** |

# **ВВЕДЕНИЕ**

В современном мире интернет стал неотъемлемой частью человеческой жизни. Поэтому для увеличения охватов и привлечения новой аудитории используются сайты лендинги (от англ. landing). В последние годы подобные сайты стали чуть ли не обязательными, для каждой уважающей себя фирмы. Лендинги также крайне эффективны при продаже цифровых товаров, позволяя раскрыть их с разных сторон.

Целью данной курсовой работы является написание сайта лендинга, на тему “Компьютерная игра Destiny 2 ”.

Целевой аудиторией являются мужчины в возрасте от 16 до28 лет.

Для достижения цели курсовой работы были поставлены следующие задачи:

1. Создать макет работы;
2. Создать основную часть Web-страницы;
3. Добавить функционал к Web-странице;
4. Проверить работоспособность Web-страницы.

Для выполнения вышеописанных задач потребовались следующие технологии:

* Язык гипертекстовой разметки - HTML;
* Каскадные таблицы стилей - CSS;
* Язык программирования JS;
* Библиотека для создания анимаций JavaScript - anime.js;
* Библиотека для создания параллакс эффектов - vanila - tilt;
* Библиотека для установки якорей waypoints
* Интерфейс JavaScript для работы с запросами HTTP - fetch API;
* Язык программирования PHP;
* API студии разработчика “BUNGIE”.

# **1. Описание сайта**

В целях демонстрации уровня знания HTML,CSS и JS был создан сайт лендинг.Помимо вышеуказанных средств разработки,по личной инициативе, были применены fetch API и PHP для обеспечения работы с серверной частью сайта. Для упрощения работы были использованы следующие библиотеки:anime.js, waypoints,vanila-tilt. В работе отсутствуют чёрный и белые цвета, их место заняли соответствующие оттенки. Реализован кастомный скроллбар. Лендинг совместим с большинством современных браузеров и сохраняет функциональность при основных разрешениях,благодаря активному использованию flexbox и grid

**1.1 Шапка сайта**

В шапке сайта находятся:логотип, ссылки а различные сегменты сайта, выпадающий список “ПРЕИМУЩЕСТВА”, телефонная ссылка,иконка для входа в аккаунт. Сама шапка сохраняет свою позицию при скроле.

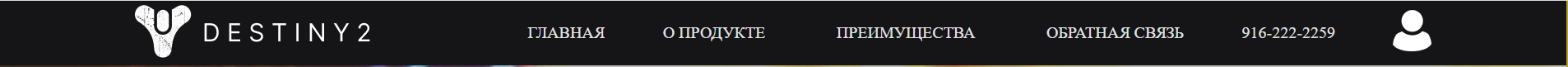


Рисунок 1. Шапка сайта

Для доступности на множественных платформа заголовок был создан с использованием flexbox.

header{display:flex;

width: 100%;

height: 8vh;

}

Телефонная ссылка по нажатию открывает окно с предложением совершить вызов по предложенному номеру, это весьма удобная функция для посетителей с устройств с функцией вызова.

При нажатии на иконку открывается диалоговое окно с формой входа и регистрации.Это диалоговое окно снабжено картинкой креста,закрывающим его,также диалоговые окна на сайте, при открытии блокируют скрол и могут быть закрыты по нажатию на затемнённую часть сайта.

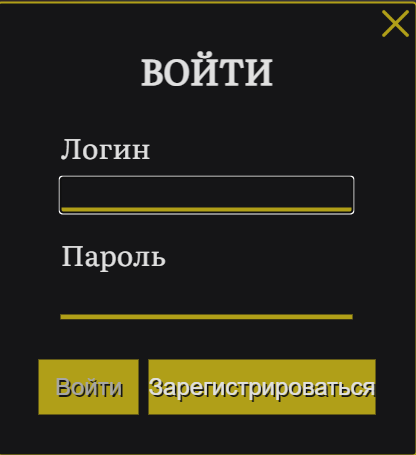


Рисунок 2. Форма входа

также как и шапка сайта диалоговые окна построены с использование flexbox:

display: flex;

flex-direction: column;

align-items: center;

Свойство flex-direction отвечает за направление основной оси блока.А allign items отвечает за расположение элементов блока на вторичной оси.

Кнопка “Войти” не активна до тех пор, пока не будут заполнены поля “Логин” и “Пароль”. При нажатии на кнопку “Войти” при помощи fetch Api из Javascript файла scripts.js форма отправляется в файл с расширением php:“logReg.php”.

await fetch('./php/logReg.php', {  
method: 'POST',  
body: new FormData(form)  
})

PHP используется для работы с серверами, а именно, для отправки и получения запросов.

Перед тем как сформировать запрос следует проверить на наличие нужных нам параметров:

if (isset($\_POST['log']))

Теперь же, можно отправить запрос на сервер с целью получения информации о пользователе:

$user\_data=mysqli\_query($link,"SELECT avatar\_url,user\_name,id FROM user where login='$login' and password=$password");

После нужно преобразовать данные в ассоциативный массив и передать их обратно в файл “scripts.js”.

Благодаря fetch API “scripts.js” уже ожидает информацию о пользователе и для её передачи нам достаточно вывести данные из logReg.php, но передавать массивы из файла в файл нельзя,для этого сначала придётся закодировать их в формат JSON:

echo (json\_encode($dec\_user\_data));

И соответственно раскодировать данные при получении:

return response.json();

Помимо этого следует занести данные в переменные сессии PHP.Это позволит не авторизовывать пользователя при переходе на другие страницы или при их обновлении.

Перед тем как начать работать с сессией её следует начать:

session\_start();

Сессию необходимо начинать перед любым обращением к серверу.

Теперь можно занести необходимые данные:

$\_SESSION['user\_id']=$dec\_user\_data['id'];

$\_SESSION['user\_name']=$dec\_user\_data['user\_name'];

$\_SESSION['user\_avatar\_url']=$dec\_user\_data['avatar\_url'];

Теперь же когда мы получили ассоциативный массив следует авторизовать пользователя,а именно вывести его имя и аватар.Для этого нужно обратится к блоку в котором находится иконка открывающая окно Входа/регистрации Сделать это мы можем используя через Javascript исрользуя объект DOM .В нём находятся изображение и, пока что пустой, абзац. Теперь нужно заменить изображение и заполнить абзац данными из массива:

document.getElementById('user\_name').textContent=json['user\_name'];

document.querySelector('.account>img').src=json['avatar\_url'];

При запуске сайта происходит fetch запрос к “startup.php”возвращающий переменные сессии(если они существуют) и авторизующий пользователя.

При нажатии на кнопку “Зарегистрироваться” открывается форма для регистрации нового пользователя, между формами осуществляется анимационный переход оформленный с помощью anime.js

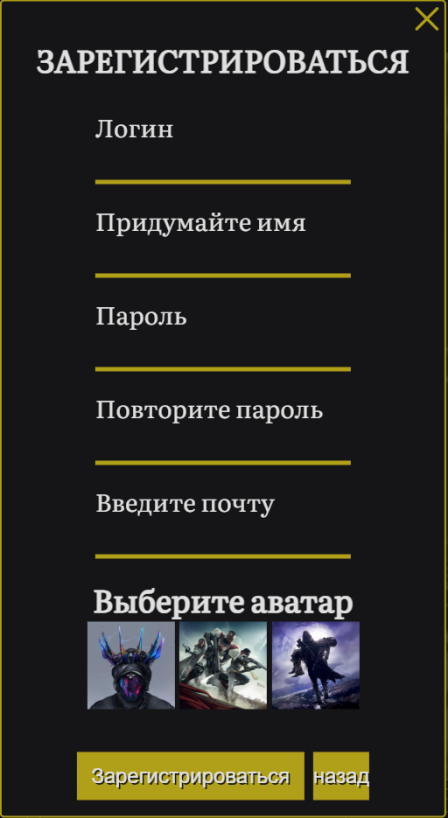
.

Рисунок 3. Форма регистрации

Форма регистрации работает схожим образом что и форма входа за исключением изменений появления дополнительного запроса к серверу в “regLog.php”:

mysqli\_query($link,"INSERT INTO `user` (`login`, `user\_name`, `password`, `email`, `avatar\_url`) VALUES ( '$login', '$nickname', '$password', '$mail', '$icon') ");

Через SQL запрос мы передаём на сервер данные из нашей формы регистрации,после передаём данные в “scripts.js” и проводим с ними те же манипуляции что и с формой входа.

После успешного входа и/или регистрации, при перезагрзке сайта вход будет происходить автоматически.Для того чтобы выти ползовотелю нужно нажать на свой аватар.После этого выполнится fetch запрос который уничтожит все данные в сессии:

session\_destroy();

В случае если вход или регистрация не были произведены успешно появится сообщение об ошибке:

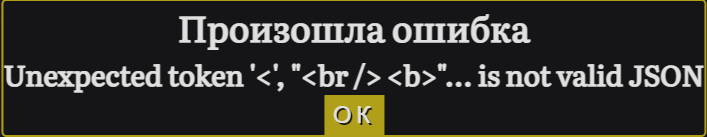


Рисунок 4. Сообщение об ошибке

Сами же формы будут очищены от введённых данных.

**1.2 Баннер**

Баннер сайта, он же первый экран который увидит пользователь также, как и весь сайт оформлен оттенках жёлтого чёрного и белого. Эти оттенки были выбраны чтобы лучше подходить визуальному стилю игры.также как и другие части сайта для добавления адаптивности сайту в этом блоке был использован flexbox:

#first\_screen{

display: flex;

justify-content: space-between;

flex-direction: column;

}

На заднем плане в на повторе и без звука находится короткое видео. Слева находится заголовок и подзаголовок являющиюся главными предложениями лендинга. Ниже находится анимированная кнопка,она пульсирует а, при наведении последний “импульс” возвращается в исходное положение.Эта анимация была создана при помощи anime.js:

anime({

targets:'#order',

outlineOffset:"20px",

outlineColor:"rgba(215, 194, 24, 0.0)",

direction: 'alternative',

loop:true,

easing:'linear',

})

Эту анимацию было решено применить для привлечения внимания пользователя к основному офферу и переходу по ссылке .

Снизу находится тёмное поле с таймером показывающим оставшееся время до конца офера намекающая на скоротечность предложение ,призывая посетителя сайта к действию, и дублирующая кнопку выше копка,при наведении она меняет цвет. Эту анимация позволяет удержать внимание пользователя.

Между двумя кнопками находится анимированный персонаж “Призрак”, двигающийся вверх-вниз и являющийся ссылкой на следующий блок.

Анимация “Призрака” также выполнена в anime.js, и призвана привлечь внимание пользователя,заставить перейти по ссылке и тем самым задержатся подольше на странице.

anime({

targets:'#next\_screen',

translateY: [90,30],

direction: 'alternate',

loop: true,

easing: 'linear',

duration:2000

})

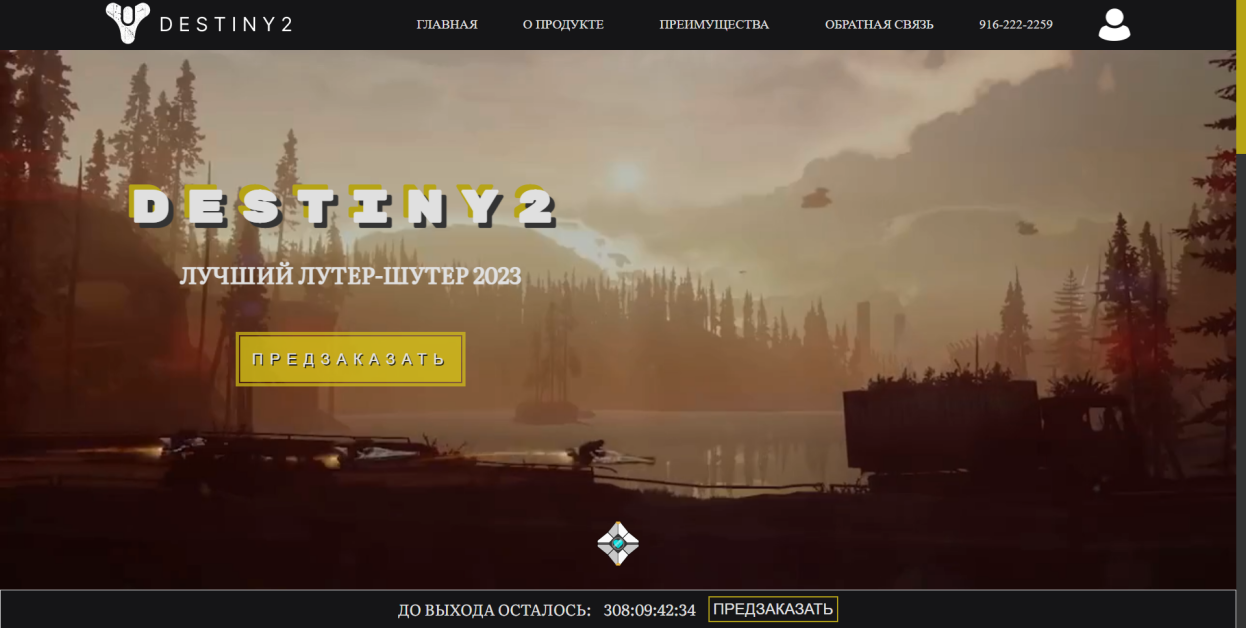


Рисунок 5. Баннер

**1.3 Блок “ОСОБЕННОСТИ DESTINY 2”**

Блок “ОСОБЕННОСТИ DESTINY 2” представляет собой горизонтальный слайдер,с отключённым скроллбаром.

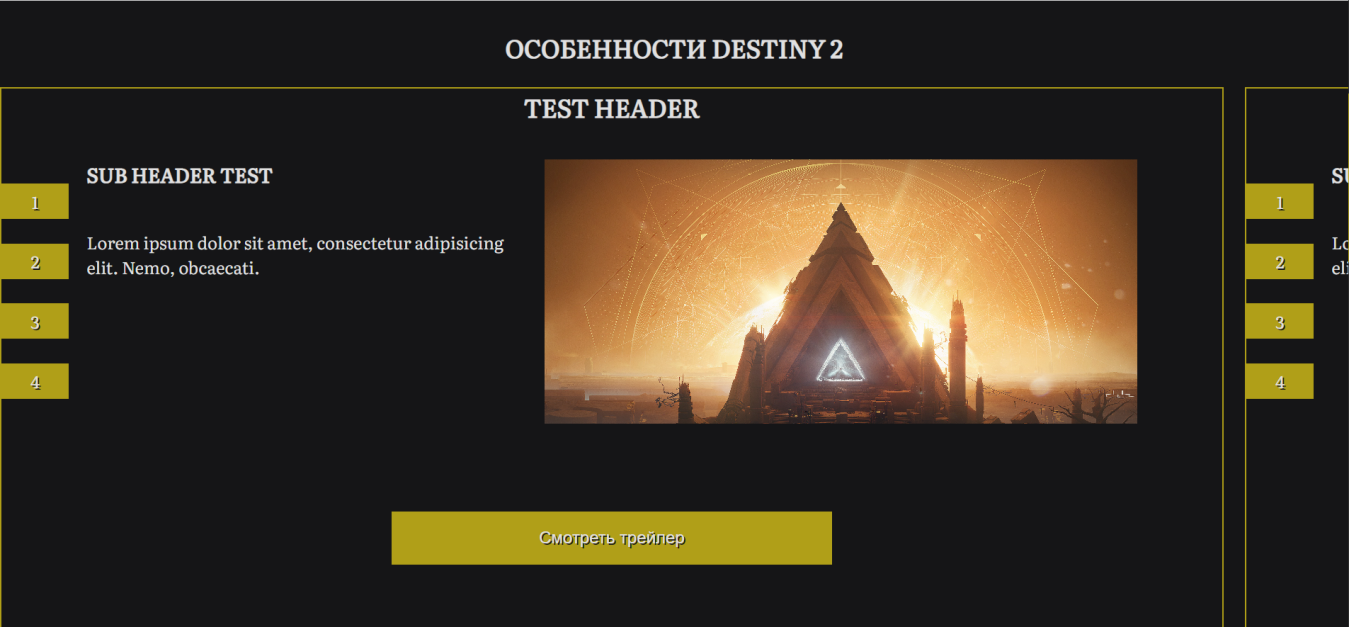


Рисунок 6.Блок “Особенности DESTINY 2”

Решение сделать этот блок горизонтальным слайдером позволило вместить большее количество контента не удлиняя сам сайт. Слева расположены закладки позволяющие быстро перемещаться между слайдами. Рядом с ними расположены подзаголовок и краткое описание, призванное заинтересовать пользователя

Справа находится иллюстрация призванная подстегнуть интерес пользователя.

Снизу расположена кнопка с призывом к действию.

Для мультиплатформенности сайта в данном сегменте было использовано CSS свойство grid.Оно позволяет сохранить оригинальную композицию элемента при изменении широты и высоты сайта.

#horizontal\_scroller{

width: 100%;

display: grid;

gap: 30px;

grid-auto-rows:80vh;

grid-auto-flow: column;

grid-auto-columns:90vw ;

scroll-snap-type: inline mandatory;

}

Свойства grid-auto-rows и grid-auto-columns отвечают за размеры и количество слайдов.

Cлайды также обладают свойством grid:

.slide{

overflow: hidden;

background-color: #151517;

border: 2px solid rgba(215, 194, 24, 0.8);

display: grid;

grid-template-columns: 0.5fr repeat(7, 1fr) 0.5fr;

grid-template-rows: repeat(8, 1fr);

grid-column-gap: 25px;

grid-row-gap: 25px;

}

Свойство grid присутствует в слайдах с той же целью что и в основном блоке.

Для более плавного перехода между слайдами было использовано свойство scroll-snap, оно позволяет браузеру доводить пользователя до следующего слайда.

#horizontal\_scroller{

scroll-snap-type: inline mandatory;

}

Свойство scroll-snap-type отвечает за поведение доводчика, конкретно эти настройки позволяет ему работать если скролл произошёл по оси x и один из слайдов оказался в поле видимости.

Самим слайдом нужно поставить отдельное свойство, которое расставит особые триггеры и вот уже к ним и будет примагничивать пользователя:

#horizontal\_scroller>\*{

scroll-snap-align: center;

}

Scroll-snap-allign определяет как будут магнититься слайды.

Несмотря на все выше описанные действия из-за того что слайдер горизонтальный для его прокрутки потребуется зажимать клавишу Shift.

С помощью JavaScript это можно исправить:

element.addEventListener('wheel', (event) => {

event.preventDefault();

element.scrollBy({

behavior: "smooth",

left: event.deltaY < 0 ? -30 : 30,

});

});

addEventListener позволяет вешать на определённые действия выполнение разнообразных функций например,вышеописанный код отлавливает любой скролл совершённый внутри горизонтального скроллера, после заменяет оригинальный скрол, своим, горизонтальным, что позволит пользователю скролить контент горизонтально не зажимая Shift.

Библиотека waypoints позволяет расставлять по сайту свои собственные якоря, которые при поподании в поле зрения пользователя будут запускать необходимые функции в том числе анимации anime.js:

let slide1=new Waypoint({

element:document.getElementById(`horizontal\_scroller`),

handler:function(){

anime({

targets:'#bookmarks\_sl1',

translateX: [-300,0],

opacity: [0,1],

easing: 'linear',

duration:500

})

this.destroy()

}

offset"90%"

})

Каждый элемент в слайдере отдельно анимируется при попадании в поле зрения,после первого срабатывания якорь уничтожается, при помощи команды this.destroy().Свойство offset отвечает за то сколько процентов объекта должно попасть в поле зрения пользователя перед срабатыванием, тем больше процентов в offset тем меньше процентов блока требуется для срабатывания. По умолчанию считается только вертикальное расположение , поэтому для слайдов отличных от первого следует добавить код:

context:document.getElementById('horizontal\_scroller'),

horizontal: true

# **1.4 Блок “ИСТОРИИ НА КАЖДОМ ШАГУ”**

Блок “ИСТОРИИ НА КАЖДОМ ШАГУ” состоит из отдельных блоков карточек

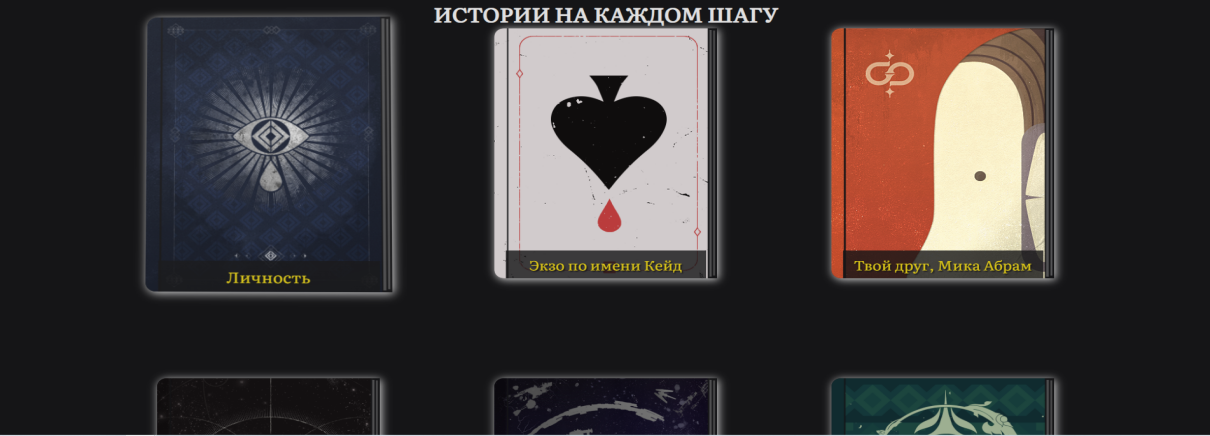


Рисунок 7. Блок “ИСТОРИИ НА КАЖДОМ ШАГУ”

Для адаптивности сайта карточки находятся в общем блоке к котрому применён flexbox:

#card\_holder{

display: flex;

justify-content: center;

flex-wrap: wrap;

gap: 180px;

width: 100%;

}

Свойство justify-content определяет расположение дочерних блоков относительно основной оси, gap расстояние между блоками, flex-wrap обеспечивает перенос блока на новую строку в случае если ему не хватает места на этой.

Также на карточках реализован паралакс эффект при помощи библиотеки tilt.js

VanillaTilt.init(document.querySelectorAll('.card'),{

glare:true,

reverse:true,

"max-glare":0.2,

scale:1.1,

});

Содержимое же получается при помощи Bungie.API.Обращаясь к нему через fetch мы получаем ассоциативный объект содержащий в себе определения книг с историями:

fetch('https://www.bungie.net/common/destiny2\_content/json/ru/DestinyPresentationNodeDefinition-88e2ca87-7551-4503-a5b5-2527c4531503.json',{}).then(response=>{return response.json()});

Ключами возвращаемого массива являются уникальные идентификаторы, значением же является также ассоциативный массив :

blacklisted: false

categoryScoreUnlockValueHash: 0

Children:…

disableChildSubscreenNavigation: false

displayProperties:…

displayStyle: 0

hash: 2637550529

index: 1119

maxCategoryRecordScore: 0

nodeType: 1

objectiveHash: 1497013875

originalIcon:"/common/destiny2\_content/icons/9300a013bff7274e6c74569d2819953c.png"

Интересными в контексте блока являются три Children,и displayProperties в первом содержатся идентификаторы записей содержащие внутри себя название страницы и идентификатор самой страницы с текстом.Внутри же displayProperties находится оригинальная, не сжатая для интерфейса игры картинка,которая и поставлена на фон

Поэтому для заполнения карточек потребуются 3 ассоциативных массива: DestinyPresentationNodeDefinition, DestinyRecordDefinition, DestinyLoreDefinition.Для упрощения работы функции по получению и картинки, заголовка и самого текста были вынесены отдельно:

function getName(id) {

return lore\_books[id]['displayProperties']['name'];

}

function getImage(id){

return `https://www.bungie.net${lore\_books[id]['displayProperties']['iconSequences']['1']['frames']['0']}`

}

function OpenLore(bookId) {

document.getElementById('lore\_name').textContent=getName(bookId);

document.getElementById('lore\_content').textContent=lore[lore\_definition[lore\_books[bookId]["children"]['records']['0']['recordHash']]['loreHash']]['displayProperties']['description'];

}

После получения ассоциативных массивов через цикл всем карточкам присваиваются фоновые картинки и заголовки, а также триггер который по нажатию будет открывать модальное окно в котором и отобразится первая страница из книги.



Рисунок 8. Модальное окно с первой страницей из книги “Экзо по имени Кейд”

# **1.5 Блок “ОТЗЫВЫ”**

Блок “ОТЗЫВЫ” для поддержания адаптивности также как блок “ИСТОРИИ НА КАЖДОМ ШАГУ”.

Ы

Рисунок 9.Блок “ОТЗЫВЫ”

Информация о отзывах находятся на сервере,для того чтобы их получить отправляется fetch запрос в “reviews.php”:

fetch('../php/reviews.php').then(response=>{

return response.json();

В “reviews.php” построчно выбираются интересующие нас данные а потом складируются в массиве который после заполнения кодируется и отправляется обратно в “scripts.js”.

while ($row) {

$user\_id = $row['user\_id'];

$review\_id = $row['id'];

$full\_review = mysqli\_query($link, "SELECT user.id, user.user\_name,user.avatar\_url,reviews.content,reviews.time FROM user JOIN reviews ON user.id=$user\_id");

$review = mysqli\_fetch\_assoc($full\_review);

$likes = mysqli\_query($link, "SELECT COUNT(\*) FROM likes WHERE review\_id=$review\_id");

array\_push($review, mysqli\_fetch\_assoc($likes));

array\_push($return, $review);

$row = mysqli\_fetch\_assoc($reviews);

}

echo(json\_encode($return));

После получения начинается создание самих отзывов,с помощью DOM, происходит создание новых объектов,заполнение их информацией и добавление им классов и последующего вкладывания друг в друга а затем и в сам сайт

let user\_name=document.createElement('h3');

user\_name.textContent=review['user\_name'];

user.appendChild(user\_name);

# **1.6 Блок “Присоединяйся”**

Блок содержит в себе flexbox для того что сайт был адаптивным

display: flex;

flex-direction: column;

justify-content: space-between;

align-items: center;

Блок присоединяйся представляет собой расположенные по центру заголовок,краткое описание, и кнопку по нажатию на котороую открывается форма входа.

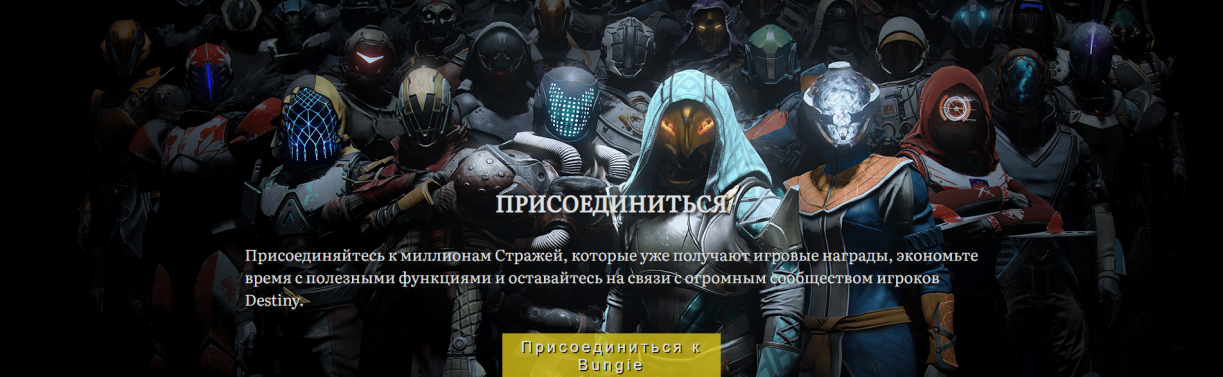


Рисунок 10.Блок “Присоединиться”

# **1.7 Подвал сайта**

Подвал не отличается от шапки за исключением добавившегося текста и иконки стрелочки при нажатии плавно скролящей наверх.

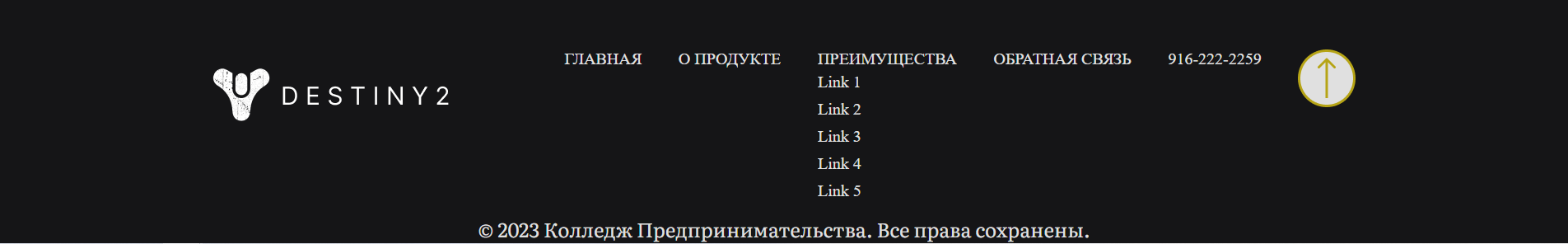


Рисунок 11.Подвал сайта

# **Заключение**

Целью моей курсовой работы было написать лендинг, на тему “Компьютерная игра Destiny 2 ”, с чем я успешно справился.Для меня же основной задачей было проверить свои знания HTML, CSS, JS, и PHP, а так же узнать в процессе что-то новое, с этим я тоже успешно справился.

Задание было выполнено следуя поставленным в введении задачам.

Выполняя поставленные задачи я решил поближе познакомится API fetch и AJAX. Проводя самостоятельное изучение этих тем я понял принцип их работы и приблизился к их освоению.Это в свою очередь позволило мне реализовать эффективную связь с MySQL сервером и Bungie.Api, исследовать последнюю мне хотелось очень давно.

Так же я хотел поэкспериментировать с библиотеками находящимся в открытом доступе.Такими как anime.js,tilt.js и waypoints. Anime.js покорил меня своей простотой и удобством а также несравненно широким функционалом.Его сочетание вместе с якорями расставленных с помощью waypoints позволил сделать горизонтальный слайдер более интересным,а tilt.js добавил уникальности карточкам

Лучшими дизайнерскими решениями я могу назвать первый экран, горизонтальный слайдер блока “ОСОБЕННОСТИ DESTINY 2” и блок “ИСТОРИИ НА КАЖДОМ ШАГУ” они понравились мне больше всего эстетически.Лучшими же идеями с точки зрения кода я могу назвать форму входа регистрации,блок “ИСТОРИИ НА КАЖДОМ ШАГУ”, отзывы. Все они были направлены на изучение внутренней части веб сайтов, на них было потрачено много усилий и времени, но сделано это было не без удовольствия .

Я не собираюсь останавливаться на достигнутом и в будущем планирую превзойти полученные сейчас результаты.