

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет инфокоммуникационных технологий

ЛАБОРАТОРНАЯ РАБОТА №2 **Взаимодействие с внешним API**

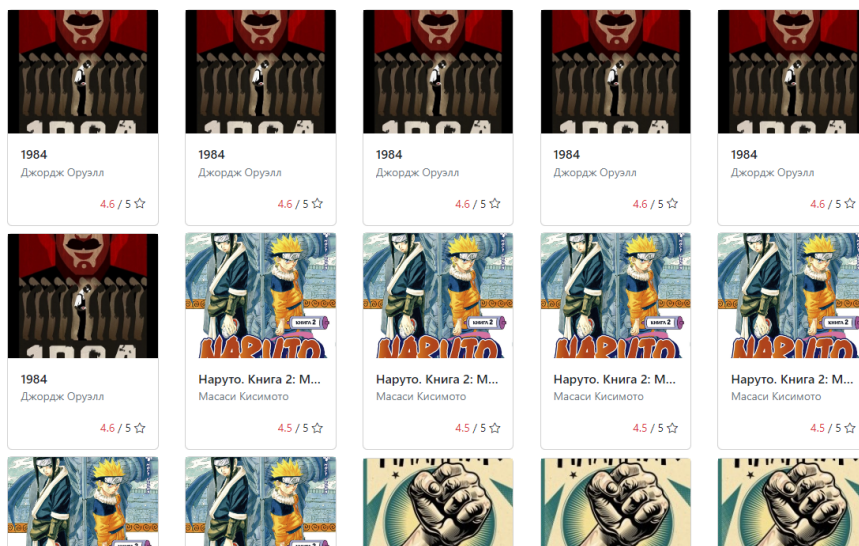
по дисциплине:
«Фронт-энд разработка»

Выполнила:
студентка III курса
группы К33401
Кормановская Д.

Санкт-Петербург
2022

Выполнение

Сортировка:



Фильтрация:

Фильтры

Показать сначала

Популярные

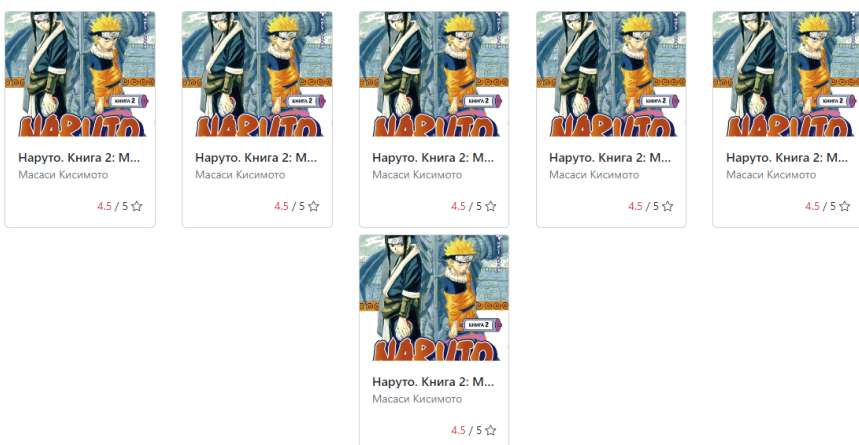
Жанр

Приключения

Объем

короткий (до 4 часов)

Подобрать



Предыдущая	1	Следующая
------------	---	-----------

Поиск:

Результаты поиска по запросу "1984"

Фильтры

Показать сначала

Популярные

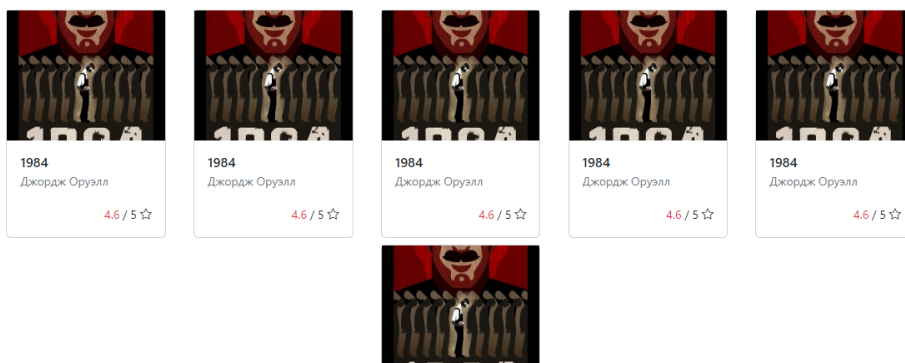
Жанр

- Любой -

Объем

- Любой -

Подобрать



Информация о книге с возможностью добавления:

1984

Джордж Оруэлл

Своеобразный антипод второй великой антиутопии XX века - "О дивный новый мир" Олдоса Хаксли. Что, в сущности, страшнее: доведенное до абсурда "общество потребления" - или доведенное до абсолюта "общество идеи"? По Оруэллу, нет и не может быть ничего ужаснее тотальной несвободы...

Жанр: [Фантастика](#)
Объем: [средний \(до 8 часов\)](#)

4.6 / 5 ☆

Уже на вашей полке

Наруто. Книга 2: Мост героев

Масаси Кисимото

Став настоящими ниндзя, Наруто, Саскэ и Сакура получают ответственное задание – охранять знаменитого строителя мостов Тадзуну из Страны Волн. На жизнь этого старика покушаются беглый синоби Дзабудза и его подопечный Хаку, обладающий невероятными способностями. Столкновение с такими опасными противниками оборачивается трагедией, когда Саскэ закрывает собой Наруто от смертоносной атаки Хаку... Кажется, участь команды Какаси уже предрешена, но в Наруто вдруг пробуждается загадочная сила... Сможет ли он переломить ход битвы?

Жанр: [Приключения](#)
Объем: [короткий \(до 4 часов\)](#)

4.5 / 5 ☆

[Буду читать](#) [Прочитано](#)

Страница пользователя:

Вы Пси

email@email.com

Прочитано книг: 3

Оставлено отзывов: 0

Книг на полке: 5

Активность

Тут графики должны быть

Достижения

Слезы это просто вода
Прочитано 30 драм

Зачем?
Ты прочитал гигантский фанфик.

Быть или не быть
вот в чем вопрос

Лента

ничего нового :)

Книжная полка

Наруто. Книга 2: Мост героев
Масаси Кисимото

прочитано

1984
Джордж Оруэлл

отложено

Бойцовский клуб
Чак Паланик

отложено

1984
Джордж Оруэлл

прочитано

Пятьдесят оттенков серого
Э.Л. Джеймс

прочитано

Изменение статуса:

1984
Джордж Оруэлл

Своеобразный антипод второй великой антиутопии XX века - "О дивный новый мир" Олдоса Хаксли. Что, в сущности, страшнее: доведенное до абсурда "общество потребления" - или доведенное до абсолюта "общество идеи"? По Оруэллу, нет и не может быть ничего ужаснее тотальной несвободы...

Жанр: [Фантастика](#)
Объем: [средний \(до 8 часов\)](#)

отложено

Обновить статус

1984
Джордж Оруэлл

Своеобразный антипод второй великой антиутопии XX века - "О дивный новый мир" Олдоса Хаксли. Что, в сущности, страшнее: доведенное до абсурда "общество потребления" - или доведенное до абсолюта "общество идеи"? По Оруэллу, нет и не может быть ничего ужаснее тотальной несвободы...

Жанр: [Фантастика](#)
Объем: [средний \(до 8 часов\)](#)

отложено

прочитано

отложено

удалить

Вход и регистрация:

Вход

Неверный адрес или пароль!

Адрес электронной почты

email@email.com

Пароль | [забыли пароль?](#)

.....

☐ Не выходить

Войти

Регистрация

Регистрация

Имя пользователя

test123

Пол

мужской

Эл. адрес

test@test.com

Подтвердите эл. адрес

test@test.com

Пароль

.....

Подтвердите пароль

.....

☒ Подписаться на новостную рассылку

☒ Согласие с правилами бла-бла

Зарегистрироваться

Примеры кода

Добавление жанров в navbar

```
function getGenre({id, name}) {
  return `<div class="p-1 mx-2"><a href="http://localhost:9999/LW1/search.html?genreId=${id}" class="text-decoration-none text-white">${name}</a></div>`
}

async function loadGenres() {
  const response = await fetch( input: "http://localhost:3000/genre")
  const responseJson = await response.json()

  for (const genre of responseJson) {
    document.querySelector( selectors: "#navGenres").innerHTML += getGenre(genre)
  }
}
```

Вход пользователя:

```
async function login(event) {
  event.preventDefault()
  const inputs = Array.from(event.target.querySelectorAll('input'))
  const loginData = {}
  // add all inputs values to loginData (except unnamed)
  for (const input of inputs) {
    if (input.name !== "") {
      loginData[input.name] = input.value
    }
  }

  // try to login via POST request
  const response = await fetch( input: 'http://localhost:3000/login', init: {
    method: "POST", body: JSON.stringify(loginData), headers: {
      'Content-Type': 'application/json'
    }
  })

  const responseJson = await response.json()
  // check mistakes
  if (responseJson === "Incorrect password" || responseJson === "Cannot find user") {
    document.querySelector( selectors: "#mistakeLogin").textContent = "Неверный адрес или пароль!"
    return
  }

  document.querySelector( selectors: "#mistakeLogin").textContent = ""
  // write user data to localStorage
  const {accessToken, user} = responseJson
  localStorage.accessToken = accessToken
  localStorage.user = JSON.stringify(user)
  location.reload()
}
```

Загрузка страницы с книгами:

```
async function loadBooks() {
  const loadparams = new URLSearchParams()
  if (params.has( name: 'timeToReadId')) {
    loadparams.set('timeToReadId', params.get('timeToReadId'))
  }
  if (params.has( name: 'genreId')) {
    loadparams.set('genreId', params.get('genreId'))
  }
  let compare = ""
  if (params.has( name: 'textSearch')) {
    compare = params.get('textSearch').toLowerCase()
  }
}
```

```

const response = await fetch( input: `http://localhost:3000/books?${loadparams.toString()}` )
const responseJson = await response.json()
const rawpages = []

for (const book of responseJson) {
  if (compare !== "" && (book.title.toLowerCase().includes(compare) || book.author.toLowerCase().includes(compare)) || compare === "") {
    rawpages.push(book)
  }
}

if (compare !== "") {
  document.getElementById( elementId: 'searchWordInfo').hidden = false
  document.getElementById( elementId: 'searchWord').textContent = compare
}

if (rawpages.length !== 0) {
  if (params.has( name: 'sort')) {
    rawpages.sort(sorts[params.get('sort')])
  }

  const pages = []
  let counter = 0
  const bound = 20

  document.querySelector( selectors: "#pages").innerHTML += getPaginationButtonHTML( name: 'prev', text: 'Предыдущая', isPrev: true, isNext: false)
  changeState( idx: 'prev')

  for (const p of rawpages) {
    const idx = Math.floor( x: counter / bound)
    if (counter % bound === 0) {
      document.querySelector( selectors: "#pages").innerHTML += getPaginationButtonHTML(idx, text: idx + 1)
      pages.push("")
    }
    pages[idx] += getBookCardHtml(p)
    counter++
  }
}

```

```

document.querySelector( selectors: "#pages").innerHTML += getPaginationButtonHTML( name: 'next', text: 'Следующая', isPrev: false, isNext: true)
if (counter < bound) {
  changeState( idx: 'next')
}

document.querySelector( selectors: "#library").innerHTML += pages[0]
localStorage.pages = pages.join(separator)
localStorage.currentpage = 0
changeState(localStorage.currentpage)
} else {
  localStorage.pages = null
  localStorage.currentpage = 0
  document.querySelector( selectors: "#pages").innerHTML += `<p>Нет соответствий :(</p>`
}
}

```

Добавление книги:

```

// Add book to user's library through book's window
async function addBook(read : boolean = false){
  const data = {}
  const userInfo = JSON.parse(localStorage.getItem( key: 'user'))
  data['userId'] = userInfo.id
  data['bookId'] = Number(document.getElementById( elementId: 'modalId').textContent)
  if (read){
    data['status'] = 'прочитано'
  }
  else {
    data['status'] = 'отложено'
  }
}

```

```

const response = await fetch( input: 'http://localhost:3000/user_book', init: {
  method: "POST",
  body: JSON.stringify(data),
  headers: {
    'Content-Type': 'application/json'
  }
})
const responseJson = await response.json()
console.log(responseJson)
document.getElementById( elementId: 'buttonsmodal').hidden = true
document.getElementById( elementId: 'already-added').hidden = false
}

```

Изменение статуса:

```

async function changeRelation() {
  const response = await fetch( input: 'http://localhost:3000/user_book/${document.getElementById( elementId: 'modalId').textContent}',
  const userbook = await response.json()
  const action = document.getElementById( elementId: 'updateoption').value

  const response2 = await fetch( input: 'http://localhost:3000/user_book/${userbook['id']}', init: {
    method: "DELETE"
  })

  if (action !== 'удалить'){
    userbook['status'] = action
    const response3 = await fetch( input: 'http://localhost:3000/user_book', init: {
      method: "POST",
      body: JSON.stringify(userbook),
      headers: {
        'Content-Type': 'application/json'
      }
    })
  })
}

document.location.reload()
}

```

Выводы

Получено общее понимание работы с внешним API и js. Используются POST, GET и DELETE запросы с использованием параметров поиска.