GET /courses Получить список курсов с возможностью сортировки

Оглавление:

История изменений

Список ссылок (ссылка) на задачи в Jira. В основном: 1 метод = 1 задача = 1 Confluence.

Связанные документы (внутренние)

Тест-кейсы, описания БД и другие статьи по проекту, которыми пользовались в ходе разработки

Общее описание

Метод `GET https://api.uverse.com/public/v1/courses` предназначен для получения списка имеющихся курсов на платформе U-Verse для:

- Потенциальных и действующих студентов для ознакомления и дальнейшей покупки курсов в системе;
- Администратор, преподаватель и модератор системы - просмотр библиотеки курсов в системе.

Доступы и ограничения

Метод должен быть доступен для:

- неавторизованных пользователей,
- авторизованных пользователей:

- студентов;
- администраторов;
- преподавателей;
- модераторов.

Запрос

Method type: GET

BaseURL: `https://api.uverse.com`

API name:

- `public`

API version: v1

Method name: /courses

Authorization:

- Без авторизации
- Студент: `Bearer Token`
- Преподаватель: `Bearer Token`
- Администратор, Модератор: `Bearer Token`

Входные данные:
HTTP params: *HEADER, QUERY*

Выходные данные:

BODY

Headers, Cookies:

- Стандартные.

Пример запроса:

```
GET https://api.uverse.com/public/v1/courses
```

query-parametrs:

1. `limit`
   Максимальное число записей на страницу.

Ограничение: целые числа от 0 до 100.

Не *обязательный*.

По умолчанию: 10.

В БД: нет

2. `offset`

   Смещение по элементам - порядковый номер элемента, с которого начинать возврат списка.

   Ограничение: целые числа больше нуля.

   Не обязательный.

   По умолчанию: 0.

   В БД: нет

3. `statusId`

   Статус курса

   Значения: UUID или числовой ID

   Ограничение: один статус, один запрос

   Не обязательный

   По умолчанию: не применяется фильтр

   в БД: да

4. `categoryId`

   Категория курса

   Значения: UUID или числовой ID

   Ограничение: не более двух категорий для одного запроса

   Не обязательный

   По умолчанию: не применяется фильтр

   В БД: да

5. `teacherId`

   Учитель

   Ограничение: может быть только один учитель

   ***Значения*:** UUID или числовой ID

   Не обязательный

   ***По умолчанию*:** не применяется.

   ***В БД*:** да

6. `tags`

Фильтр по тегам.

***Значения*:** список тегов через запятую

Ограничения: не более 5 тегов для поиска

Не обязательный

***По умолчанию*:** не применяется.

***В БД*:** да

7. `sort`

Поле и направление сортировки.

Значения:

- `title:asc` / `title:desc`
- `created_at:asc` / `created_at:desc`
- `price:asc` / `price:desc`

  Ограничения: не более двух сортировок

  Не обязательный.

  По умолчанию: `price desc`.

  В БД: нет

Пример запроса с `query` -параметрами

```
GET https://api.uverse.com/public/v1/courses?limit=10&offset=0&
teacherId=g78hj90k-lm12-nopq-3456-rstuvwxy789z&statusId=active&sort=price:desc
```

Тело запроса (Body)

Не предусмотрено

Пример ответа - Успех

Код HTTP: 200

Тело ответа (Body):

```
{
  "courses": [
    {
       "id": "92e9a899-c126-4f05-b25c-7329d15eecbd",
       "title": "Основы Python",
       "avatarUrl": "/images/courses/courseId/avatar/name.png",
       "status":{
           "id":edb37b89-563d-46c9-b26e-819a27589150,
           "title": "in_work"
       },
       "category": [
```

```
          {
            "id": "440b59ba-996c-412b-acb6-7be41362f71a",
            "title": "Програмирование"
          }
        ],
        "teacher":
            {
              "id": "a1b2c3d4-5678-90ef-1234-567890abcdef",
              "Name": "Петров Максим Геннадьевич"
            }
        "price": "9900",
        "tags": ["backend", "beginner"]
         },
     {
       "id": "550e8400-e29b-41d4-a716-446655440000",
       "title": "Веб-дизайн с нуля",
       "coverUrl": "/images/courses/courseId/avatar/name.png",
       "statusId": {
             "id": "e06bf86e-22d3-4d93-9bbc-c0319eef9f3e",
             "title": "archived"
       },
       "category": {
             "id": "a93f5a20-2fed-486e-b5fd-a16cc736d77a",
             "title": "design"
             },
       "teacher":
           {
              "id": "b2c3d4e5-6789-01fg-2345-678901abcdef",
              "Name": "Зверева Антонина Валерьевна"
           },
       "price": 14900,
       "tags": [
       "ui/ux",
       "figma"]
     }
   ],
   "pagination": {
       "total": 2,
       "limit": 10,
       "offset": 0
       }
}
```

Пример ответа - Успех с пустым полем ответа

Код HTTP: 200

Тело ответа (Body):

```
{
 "courses": [],
 "pagination": {
      "total": 0,
      "limit": 10,
      "offset": 0
      }
}
```

Пример ответа - Неправильный запрос

Код HTTP: 400

Тело ответа (Body):

1. `limit`

```
{
    "error": {
       "text": "RequestValidationError",
       "message": "В параметре "limit" ожидалось целое число от 0 до 100"
      }
}
```

2. `offset`

```
{
    "error": {
       "text": "RequestValidationError",
       "message": "В параметре "offset" ожидалось целое число больше 0"
      }
}
```

3. `statusId`

```
{
    "error": {
       "text": "RequestValidationError",
       "message": "В параметре "statusId" более одного статуса, запрос осуществляется только п
      }
}
```

4. `category`

```
{
    "error": {
       "text": "RequestValidationError",
       "message": "Параметр "categoryId" не может быть выполнен, так как выбрано более двух ка
      }
}
```

5. `teacherId`

```
{
    "error": {
       "text": "RequestValidationError",
       "message": "Параметр "teacherId" не может быть выполнен, так как выбран более одного пр
      }
}
```

6. `tags`

```
{
    "error": {
```

```
       "text": "RequestValidationError",
       "message": "Параметр "tags" не может быть выполнен, так как выбрано более пяти тегов дл
      }
}
```

7. `sort`

```
{
    "error": {
       "text": "RequestValidationError",
       "message": "Параметр "sort" не может быть выполнен, так как выбрано более двух сортиров
      }
}
```

Пример ответа - Запись не найдена

Код HTTP: 404

Тело ответа (Body):

1. `statusId`

```
{
  "error": {
       "text": "Not Found",
       "message": "Статус не найден"
    }
}
```

2. `category`

```
{
  "error": {
       "text": "Not Found",
       "message": "Категория не найдена"
    }
}
```

3. `teacherId`

```
{
  "error": {
       "text": "Not Found",
       "message": "Преподаватель не найден"
    }
}
```

4. `tags`

```
{
  "error": {
       "text": "Not Found",
       "message": "Тэги не найдены"
    }
}
```

Пример ответа - Сервер не доступен

Код HTTP: 503

Тело ответа (Body):

```
{
  "error": {
    "text": "Service Unavailable",
    "message": "Сервер не доступен, попробуйте еще раз"
   }
}
```

Требования к реализации

Алгоритм работы:

1. Пользователь запрашивает список курсов.
2. **Frontend** вызывает метод `GET` `https://api.uverse.com/public/v1/courses`
3. **Backend** проверяет есть ли *query-параметры* в запросе:
   a. Если *query-параметры* отсутствуют, переходим к пункту 5 алгоритма работы
   b. Если *query-параметры* есть, переходим к пункту 4 алгоритма работы
4. **Backend** проверить query-параметры на соответствия ограничениям.
   Если хотя бы один из них не соответствует ограничения, то вернуть ошибку HTTP 400 с соответствующем текстом, описанным выше.
5. **Backend** отправляет SQL запрос к **БД** для получения данных.
   a. Если *query-параметров* нет SQL запрос:
      По умолчанию стоит сортировка *price desc*

```
SELECT
  cs.name,
  cs.description,
  cs.duration,
  cs.price,
  cs.category_id,
  cs.state_id,
  emp.first_name,
  emp.last_name,
  emp.middle_name
FROM course as cs
JOIN course_employees as ce
ON cs.id = ce.course_id
JOIN employees as emp
ON ce.employees_id = emp.id
WHERE emp.role_id = 3
ORDER BY price desc;
```

   b. Если query-параметры есть SQL запрос

```
SELECT
  cs.name,
  cs.description,
  cs.duration,
  cs.price,
  cs.requirements_for_students,
  cs.category_id,
  cs.state_id,
  emp.first_name,
  emp.last_name,
  emp.middle_name
FROM course as cs
JOIN course_employees as ce
ON cs.id = ce.course_id
JOIN employees as emp
ON ce.employees_id = emp.id
where emp.id = {{teacher_id}};
```

```
SELECT
  cs.name,
  cs.description,
  cs.duration,
  cs.price,
  cs.requirements_for_students,
  cs.category_id,
  cs.state_id,
  emp.first_name,
  emp.last_name,
  emp.middle_name
FROM course as cs
JOIN course_employees as ce
ON cs.id = ce.course_id
JOIN employees as emp
ON ce.employees_id = emp.id
where state_id = {{state_id}};
```

```
SELECT
  cs.name,
  cs.description,
  cs.duration,
  cs.price,
  cs.requirements_for_students,
  cs.category_id,
  cs.state_id,
  emp.first_name,
  emp.last_name,
  emp.middle_name
FROM course as cs
JOIN course_employees as ce
ON cs.id = ce.course_id
JOIN employees as emp
ON ce.employees_id = emp.id
where category_id = {{category_id}};
```

```
SELECT
  cs.name,
  cs.description,
  cs.duration,
```

```
  cs.price,
  cs.requirements_for_students,
  cs.category_id,
  cs.state_id,
  emp.first_name,
  emp.last_name,
  emp.middle_name,
  tags.name
FROM course as cs
JOIN course_employees as ce
ON cs.id = ce.course_id
JOIN employees as emp
ON ce.employees_id = emp.id
JOIN tags_course as tc
ON cs.id = tc.course_id
JOIN tags
ON  tc.tags_id = tags.id
where tags.id ={{tags_id}} and emp.role_id = 3;
```

6. **БД** возвращает данные **Backend**.
7. **Backend** формирует JSON ответа, пример описан выше.
8. **Backend** возвращает JSON на **Frontend** с кодом ответа HTTP 200
   a. **a.** Если с выбранными query-параметрами записей не найдено, то вернуть HTTP 200 с []+pagination, пример ответа описан выше.
9. **Frontend** отображает список курсов пользователю согласно выставленным параметрам `limit` и `offset`.
   a. Если параметры пагинации не были выставлены, возвращаются по умолчанию `limit=10`, `offset=0`

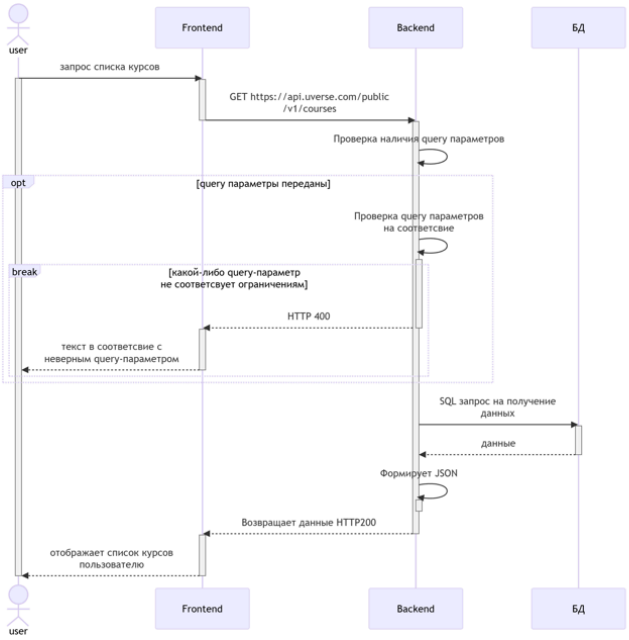
Обработка исключительных ситуаций:

1A. Внутренняя ошибка - возможна на любом шаге.
В процессе работы алгоритма не удается подключиться к БД или происходит внутренняя ошибка. Вернуть пользователю код ошибки HTTP 503 с соответствующим текстом “Сервис временно недоступен”.

код для mermaid диаграммы получения списка курсов

```
sequenceDiagram
actor user
user->>Frontend: запрос списка курсов
activate user
activate Frontend
Frontend->>Backend: GET https://api.uverse.com/public<br/>/v1/courses
deactivate Frontend
activate Backend
Backend->>Backend: Проверка наличия query параметров
opt query параметры переданы
Backend->>Backend: Проверка query параметров<br/> на соответсвие
```

```
activate Backend
break какой-либо query-параметр<br/> не соответсвует ограничениям
Backend-->>Frontend: HTTP 400
deactivate Backend
activate Frontend
Frontend-->>user: текст в соответсвие с<br/> неверным query-параметром
deactivate Frontend
end
end
Backend->>БД: SQL запрос на получение<br/> данных
activate БД
БД-->>Backend: данные
deactivate БД
Backend->>Backend: Формирует JSON
activate Backend
deactivate Backend
Backend-->>Frontend: Возвращает данные HTTP200
deactivate Backend
activate Frontend
Frontend-->>user: отображает список курсов<br/> пользователю
deactivate Frontend
deactivate user
```



Маппинг данных

Я добавила бы колонки - может быть null или не может, default, mask (нужно ли валидировать значение по какой-то маске, например, uuidv4)

Параметр в JSON	Описание / комментарий, требования к	Обяз.	Тип данных JSON	В БД

	преобразованию после извлечения из БД или обработки			
`courses`	Список курсов	✅	`Array<Objects>`	
`courses.id`	Уникальный идентификатор курсов	✅	`String`	course.id
`courses.title`	Наименование курсов	✅	`String`	course.title
`courses.avatarUrl`	Ссылка на аватар курса	✅	`String`	course.avatarUrl
`courses.status`	Статус курса	✅	`object`	
`courses.status.id`	Уникальный идентификатор статуса	✅	`String`	course.status
`courses.status.name`	Наименование курса	✅	`String`	status.name
`courses.category`	Категории курса	✅	`Array<Objecs>`	course_category

`courses.category.id`	Уникальный идентификатор	✅	`String`	category.id
`courses.category.name`	Наименование категории	✅	`String`	category.name
`courses.teacher`	Преподаватель	✅	`object`	
`courses.teacher.id`	Уникальный идентификатор преподавателя	✅	`String`	teacher.id
`courses.teacher.name`	ФИО преподавателя	✅	`String`	Конкатенация (=склейка): teacher.firstname+” “+teacher.middlename+” “+teacher.lastname
`courses.price`	Цена	✅	`number`	course. price
`courses.tags`	Тэги	⛔	`Array`	courses.tags
`pagination`	Пагинация	✅	`object`	

`total`	всего курсов по запросу	+	`number`	— (вычисляемое)
`limit`	Лимит курсов на странице	+	`number`	— (параметр запроса)
`offset`	Смещение пагинации	+	`number`	— (параметр запроса)