САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа 4

Выполнил:

Беломытцев Андрей

К3339

Проверил:
Добряков Д. И.

Санкт-Петербург

2025 г.

Задача

Тестирование API средствами Postman

- реализовать тестирование API средствами Postman;
- написать тесты внутри Postman.

Ход работы

Были написаны следующие тесты с использованием Postman:

POST Login User

```
pm.test('Get token from response', function () {
    pm.response.to.have.status(200);
    const jsonData = pm.response.json();
    pm.response.to.have.jsonBody('token');
});
```

GET Get User

```
const schema = {
  type: 'object',
  required: ['id', 'username', 'email', 'timeCreate', 'about', 'channels',
'reviews'],
  properties: {
    id: { type: 'integer' },
    username: { type: 'string' },
    email: { type: 'string', format: 'email' },
    timeCreate: { type: 'string', format: 'date-time' },
    about: { type: ['string', 'null'] },
    channels: {
      type: 'array',
      items: {
          type: 'string'
      }
    },
    reviews: {
      type: 'array',
      items: {
          type: 'integer'
      }
    }
  }
};

pm.test('Response matches user schema', function () {
  pm.response.to.have.status(200);
  pm.expect(pm.response.json()).to.be.an('object');
  pm.expect(tv4.validate(pm.response.json(), schema)).to.be.true;
});
```

GET Get Role

```
const schema = {
  type: 'object',
  required: ['id', 'name', 'users'],
  properties: {
    id: {
      type: 'integer',
      minimum: 1
    },
    name: {
      type: 'string',
      minLength: 1
    },
    users: {
      type: 'array',
      items: {
        type: 'integer',
        minimum: 1
      }
    }
  }
};

pm.test('Response matches role schema', function () {
  pm.response.to.have.status(200);
  pm.expect(tv4.validate(pm.response.json(), schema)).to.be.true;
});
```

GET Get Channel

```
const schema = {
  type: 'object',
  required: [
    'id', 'url', 'title', 'views', 'subs', 'videos',
    'lang', 'iconDefault', 'iconMedium', 'iconHigh',
    'description', 'isApproved', 'timeCreate', 'timeUpdate',
    'category', 'theme', 'videosList', 'reviews'
  ],
  properties: {
    id: {
      type: 'string',
      pattern: '^UC[a-zA-Z0-9_-]{22}$'
    },
    url: {
      type: 'string',
      format: 'uri'
    },
    title: {
      type: 'string'
    },
    views: {
      type: 'string',
    },
    subs: {
      type: 'integer',
      minimum: 0
    },
    videos: {
      type: 'integer',
      minimum: 0
    },
```

```
    lang: {
      type: 'string',
      pattern: '^[a-z]{2}$'
    },
    iconDefault: {
      type: 'string',
      format: 'uri'
    },
    iconMedium: {
      type: 'string',
      format: 'uri'
    },
    iconHigh: {
      type: 'string',
      format: 'uri'
    },
    description: {
      type: 'string'
    },
    isApproved: {
      type: 'boolean'
    },
    timeCreate: {
      type: 'string',
      format: 'date-time'
    },
    timeUpdate: {
      type: 'string',
      format: 'date-time'
    },
    category: {
      type: 'integer',
      minimum: 0
    },
    theme: {
      type: 'integer',
      minimum: 0
    },
    videosList: {
      type: 'array',
      items: {
        type: 'string',
        pattern: '^[a-zA-Z0-9_-]{11}$'
      }
    },
    reviews: {
      type: 'array',
      items: {
        type: 'integer',
        minimum: 0
      }
    }
  }
};

pm.test('Response matches channel schema', function () {
  pm.response.to.have.status(200);
  pm.expect(tv4.validate(pm.response.json(), schema)).to.be.true;
});
```

GET Get Category

```
const schema = {
  type: 'object',
  required: ['id', 'name', 'channels'],
  properties: {
    id: {
      type: 'integer',
      minimum: 1
    },
    name: {
      type: 'string',
      minLength: 1
    },
    channels: {
      type: 'array',
      items: {
        type: 'string',
        pattern: '^UC[a-zA-Z0-9_-]{22}$'
      }
    }
  }
};

pm.test('Response matches category schema', function () {
  pm.response.to.have.status(200);
  pm.expect(tv4.validate(pm.response.json(), schema)).to.be.true;
});
```

GET Get Theme

```
const schema = {
  type: 'object',
  required: ['id', 'name', 'channels'],
  properties: {
    id: {
      type: 'integer',
      minimum: 1
    },
    name: {
      type: 'string',
      minLength: 1
    },
    channels: {
      type: 'array',
      items: {
        type: 'string',
        pattern: '^UC[a-zA-Z0-9_-]{22}$'
      }
    }
  }
};

pm.test('Response matches theme schema', function () {
  pm.response.to.have.status(200);
  pm.expect(tv4.validate(pm.response.json(), schema)).to.be.true;
});
```

GET Get Review

```
const schema = {
```

```
  type: 'object',
  required: ['id', 'rate', 'text', 'timeCreate', 'channelId', 'userId'],
  properties: {
    id: {
      type: 'integer',
      minimum: 1
    },
    rate: {
      type: 'integer',
      minimum: 1,
      maximum: 5
    },
    text: {
      type: 'string',
      minLength: 1
    },
    timeCreate: {
      type: 'string',
      format: 'date-time'
    },
    channelId: {
      type: 'string',
      pattern: '^UC[a-zA-Z0-9_-]{22}$'
    },
    userId: {
      type: 'integer',
      minimum: 1
    }
  }
};

pm.test('Response matches review schema', function () {
  pm.response.to.have.status(200);
  pm.expect(tv4.validate(pm.response.json(), schema)).to.be.true;
});
```

GET Get Video

```
const schema = {
  type: 'object',
  required: ['id', 'title', 'publishedAt', 'thumbnail', 'length', 'views',
'description', 'channelId'],
  properties: {
    id: {
      type: 'string',
      pattern: '^[a-zA-Z0-9_-]{11}$'
    },
    title: {
      type: 'string',
      minLength: 1
    },
    publishedAt: {
      type: 'string',
      format: 'date-time'
    },
    thumbnail: {
      type: 'string',
      format: 'uri'
    },
    length: {
```

```
      type: ['integer', 'null'],
      minimum: 0
    },
    views: {
      type: ['integer', 'null'],
      minimum: 0
    },
    description: {
      type: 'string'
    },
    channelId: {
      type: 'string',
      pattern: '^UC[a-zA-Z0-9_-]{22}$'
    }
  }
};

pm.test('Response matches video schema', function () {
  pm.response.to.have.status(200);
  pm.expect(tv4.validate(pm.response.json(), schema)).to.be.true;
});
```

Вывод

В результате было реализовано тестирование API средствами Postman и написаны тесты внутри Postman. Все тесты были пройдены.