**Привет всем, кто добрался до этого этапа! Вы уже проделали огромную работу, осталось совсем немного!**

[**Макет тут**](https://www.figma.com/file/d8LhhLjMkaTfPvAcYQULNv/Library---students-file?node-id=3604%3A32183&t=jrcpqBLPtm08moiB-0)

[**API тут**](https://docs.google.com/document/d/1B5DmTgpTDSlyUpWu4EbyHxi_hyccoBSP4ifX6Qr3LOQ/edit)

## 

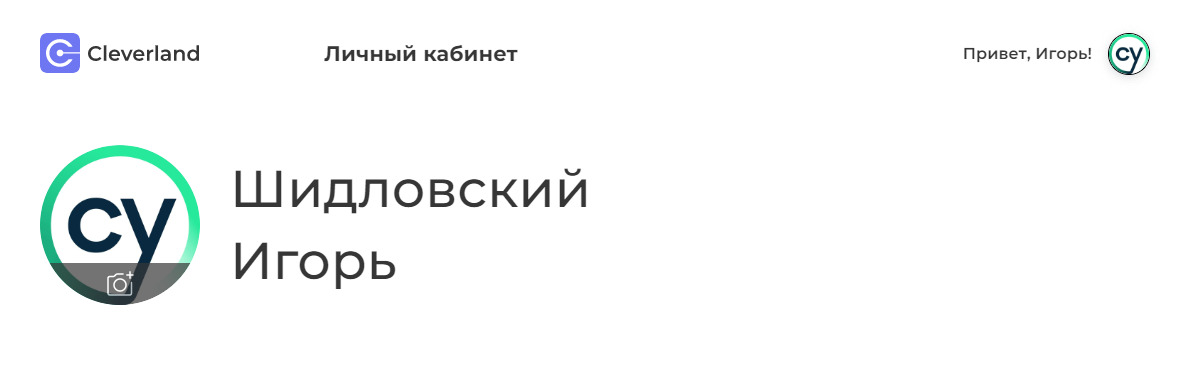
Никто еще не задумывался, а как же посмотреть те книги которые мы забронировали в прошлом спринте, или историю книг, как вернуть книгу или посмотреть до какой даты она взята? Вот как раз для этого и нужен личный кабинет пользователя. Всё это и многое другое в спринте 7 - приступаем!

1. **Получение данных пользователя.**

Т.к. мы уже авторизованные пользователи нашей библиотеки, то, конечно же можно получить данные этого пользователя

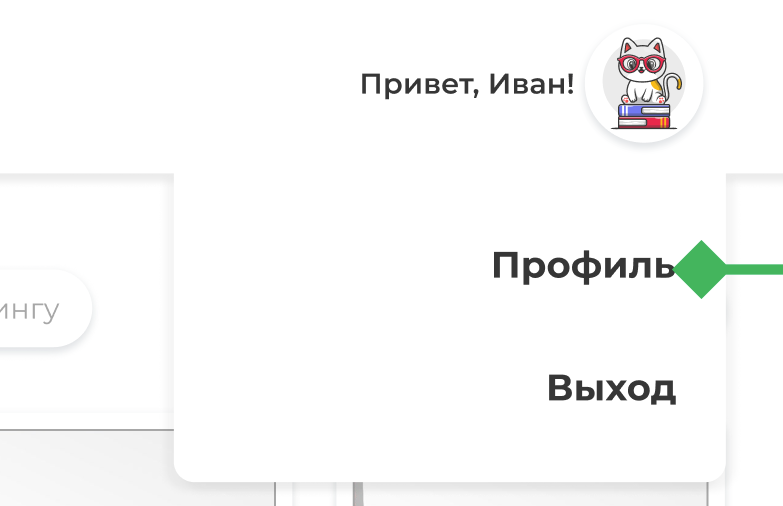
URL: /api/users/me  
метод: GET  
[Подробнее API](https://docs.google.com/document/d/1B5DmTgpTDSlyUpWu4EbyHxi_hyccoBSP4ifX6Qr3LOQ/edit)

Соответственно отображаем имя в header-e



Для перехода на страницу профиля реализовать выпадающее меню. По роутингу для страницы профиля используйте ***/profile.***  При наведении ***hover***  на аккаунт в heder появляется меню.

P.S.: если реализовали по другому - тест не должен упасть здесь.

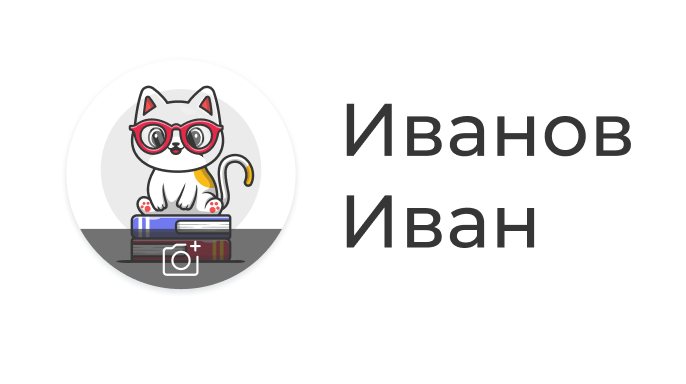


На пункт “Профиль” добавить ***data-test-id=’profile-button’***

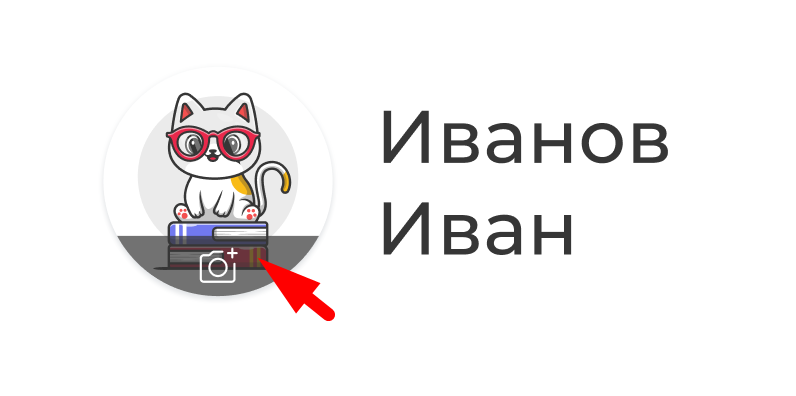
Для главной страницы добавить ***data-test-id='main-page'***

1. **Смена аватарки**

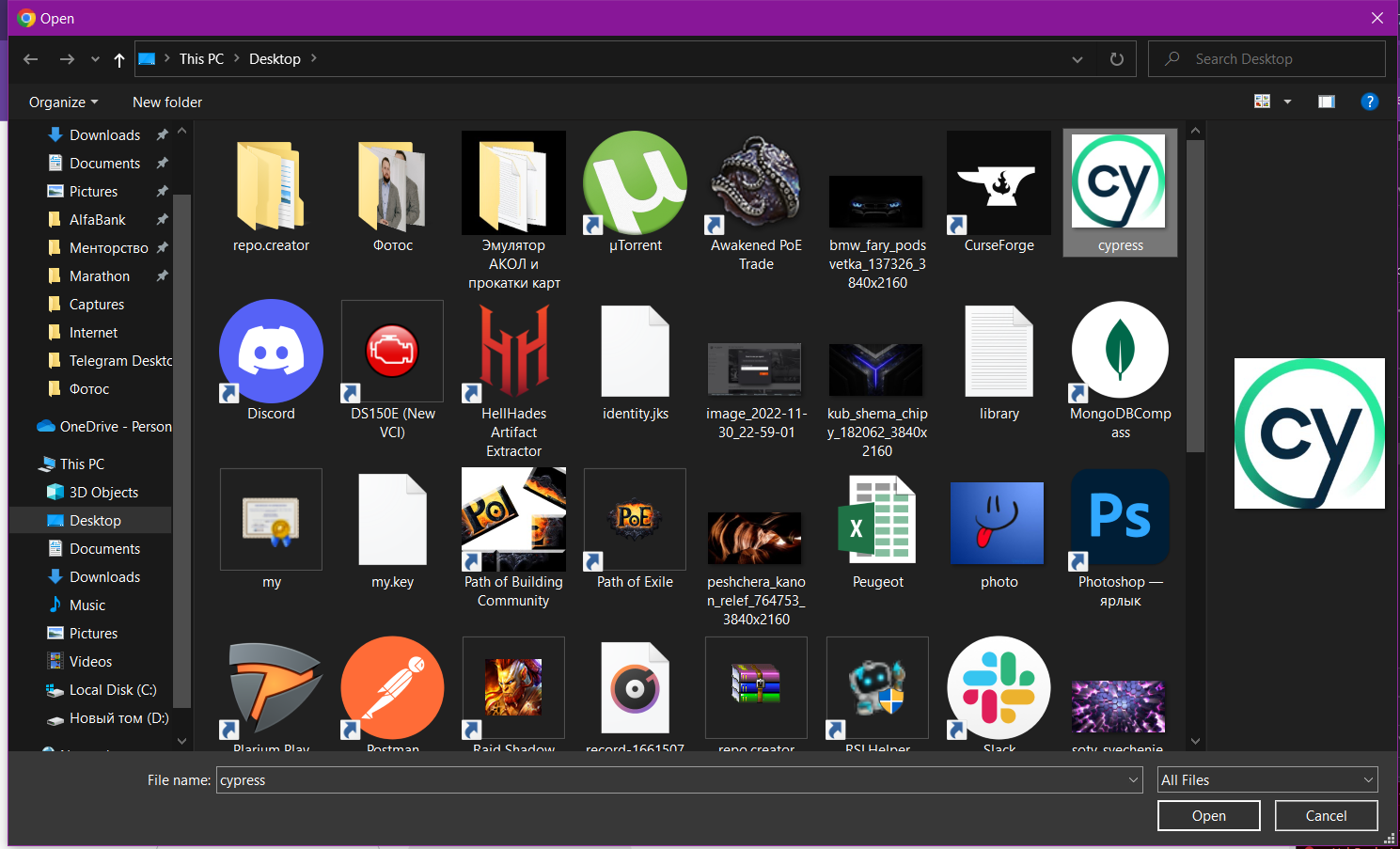
Если у пользователя нет фото должна быть картинка заглушка с котиком



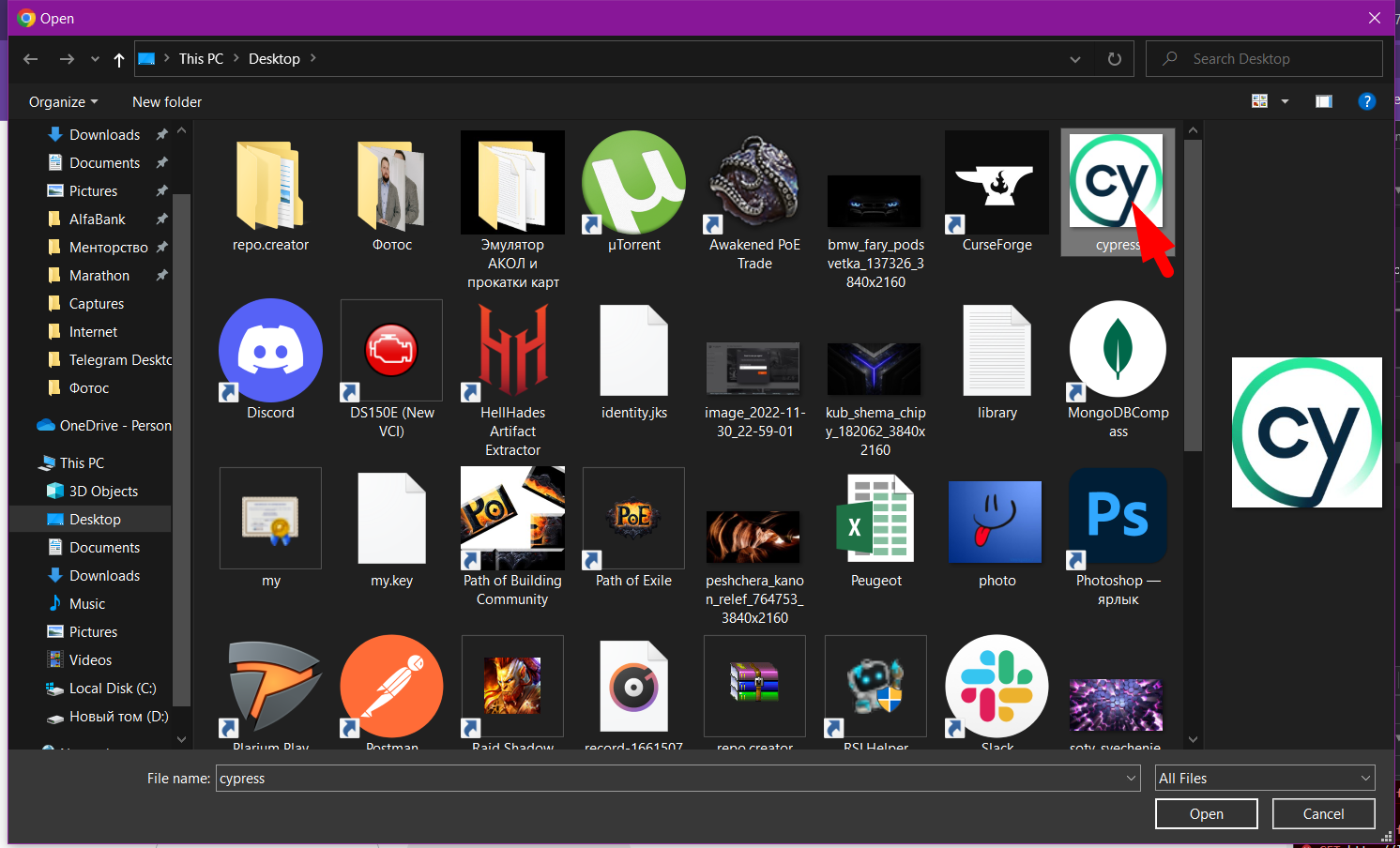
Небольшой алгоритм смены аватарки:

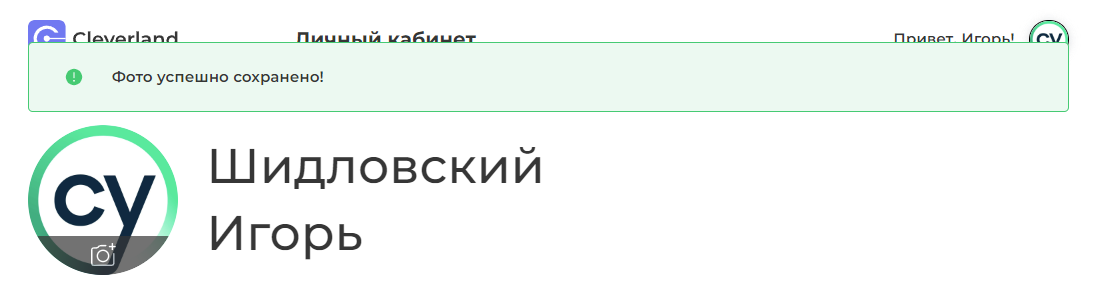
Пользователь кликает на значок, или саму картинку -   


появляется стандартный проводник, для выбора файла



Выбираем картинку и при нажатии на “Open” -



происходит загрузка и смена картинки у пользователя  


Для того чтобы всё получилось придется сначала загрузить изображение на сервер:

**Загрузка изображения**:

URL: /api/upload  
метод: POST

[Подробнее API](https://docs.google.com/document/d/1B5DmTgpTDSlyUpWu4EbyHxi_hyccoBSP4ifX6Qr3LOQ/edit)

Когда изображение успешно загружено - нужно “привязать” его к конкретному пользователю -

**Изменение аватара:**

Взять поле id из ответа загруженного изображения: "id": 6,

URL: /api/users/{userId}  
метод: PUT

[Подробнее API](https://docs.google.com/document/d/1B5DmTgpTDSlyUpWu4EbyHxi_hyccoBSP4ifX6Qr3LOQ/edit)

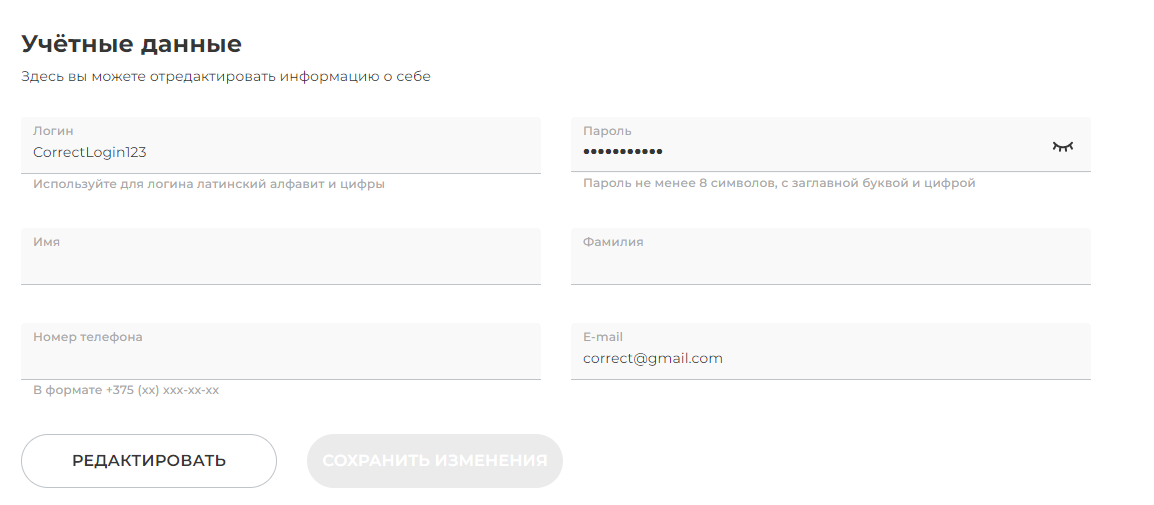
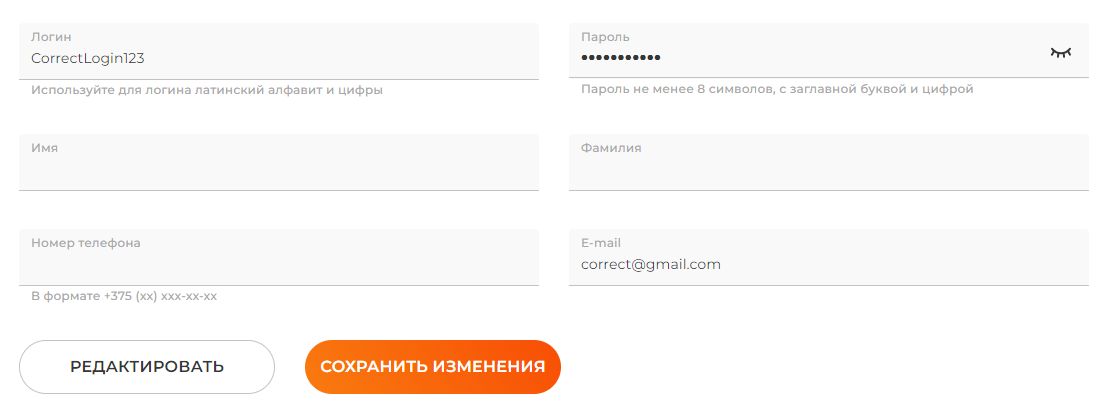
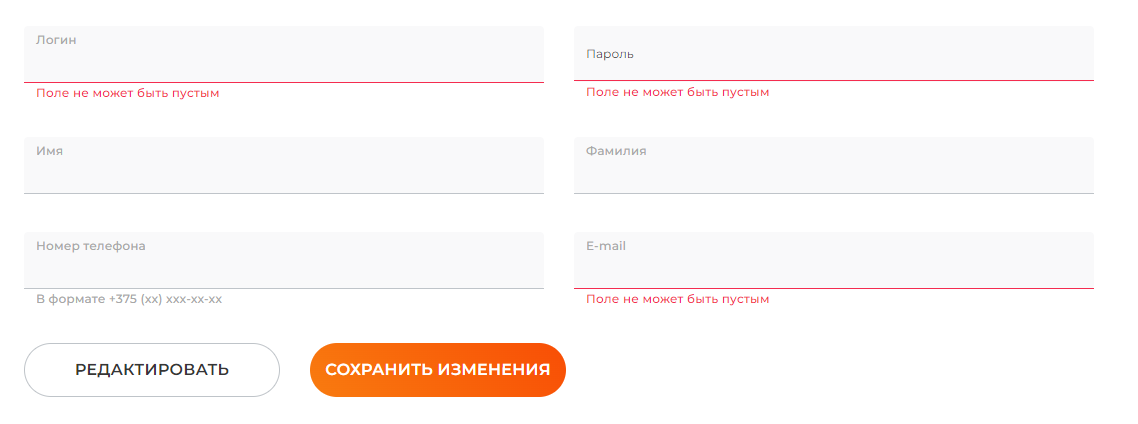
Для обертки блока аватара с именем и фамилией добавить ***data-test-id='profile-avatar'***

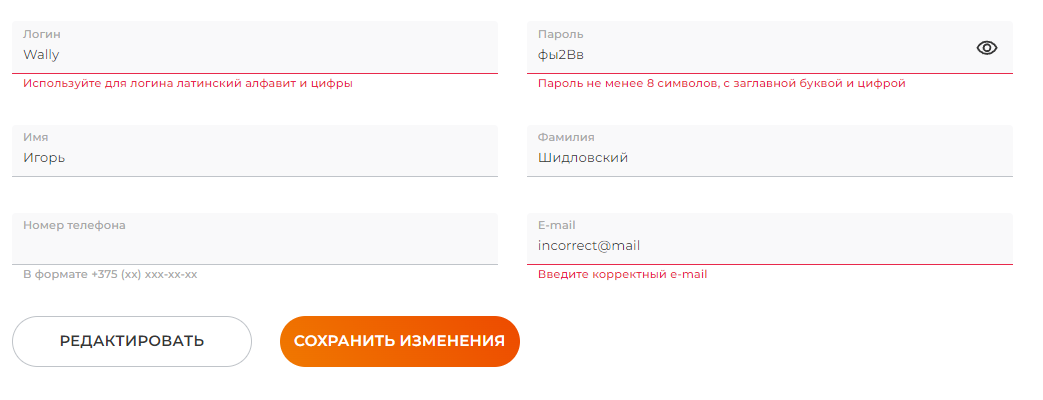
1. **Редактирование данных пользователя**

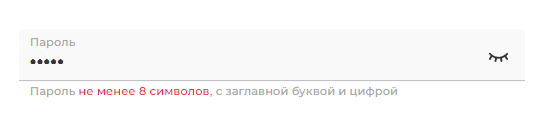
Предусмотрено редактирование профиля пользователя, с валидацией обязательных полей.

URL: /api/users/{userId}  
метод: PUT

[Подробнее API](https://docs.google.com/document/d/1B5DmTgpTDSlyUpWu4EbyHxi_hyccoBSP4ifX6Qr3LOQ/edit)

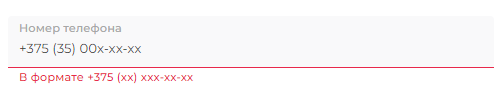
* Кнопка “Редактировать” по умолчанию активна, кнопка “Сохранить изменения” - заблокирована, все поля заблокированы.****
* При нажатии на “Редактировать” все поля становятся доступны к редактированию, кнопка “Сохранить изменения” становится активной
* Обязательные поля “Логин”, “Пароль” и “E-mail”- валидация onBlur - валидация аналогична валидации тех же полей при регистрации



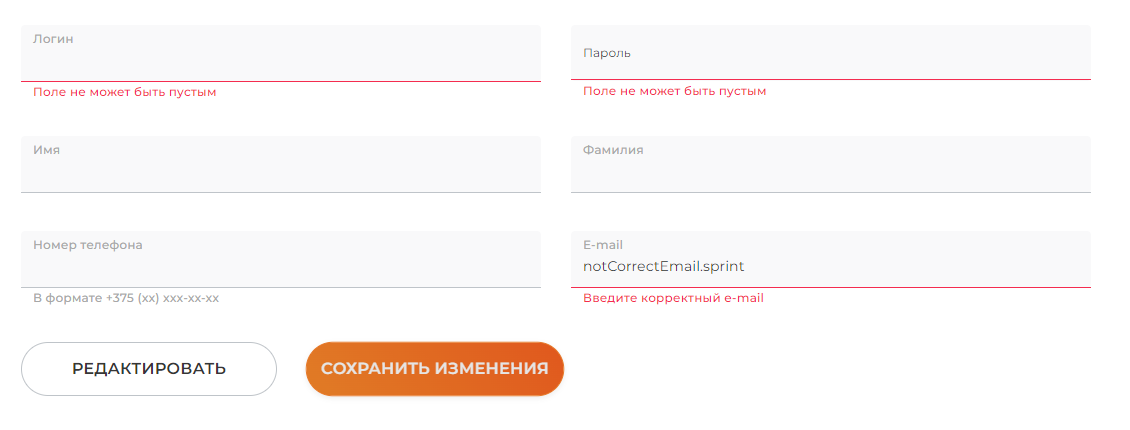
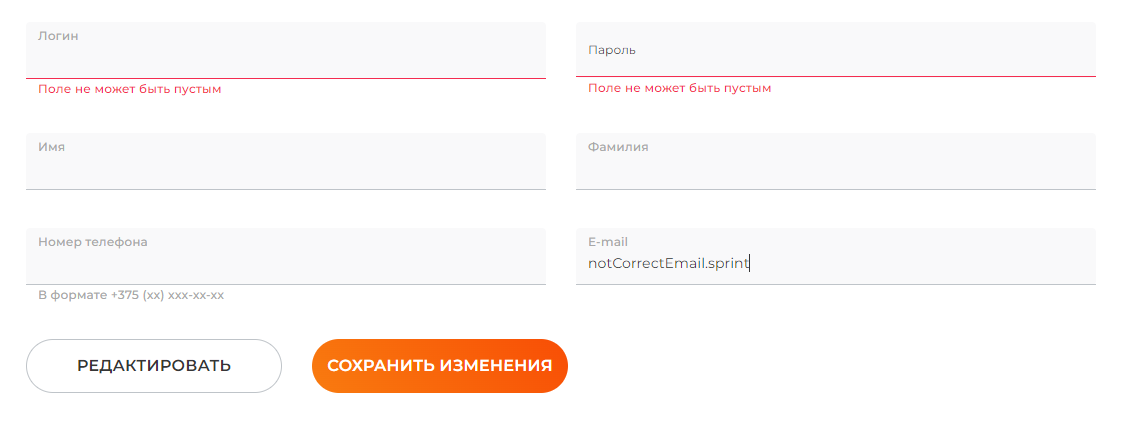
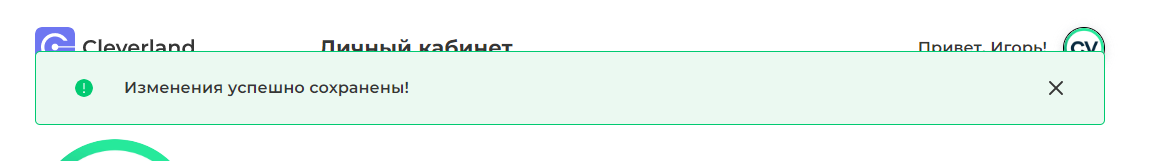
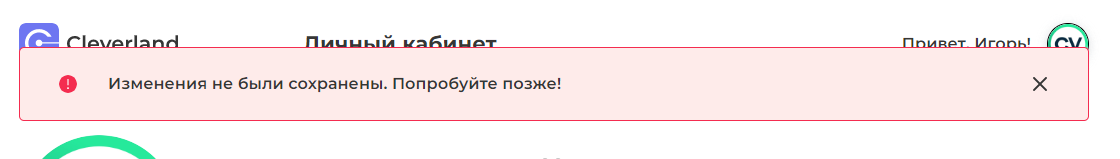


* Валидация номера телефона должна присутствовать хоть поле и не является обязательным, логично, чтобы можно было отправлять адекватные номера телефонов





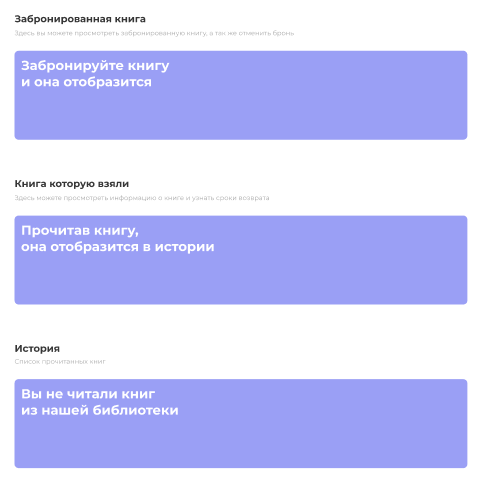
формат телефона номера белорусских операторов

* При нажатии на кнопку “Сохранить изменения” происходит проверка полей - если есть не валидные поля - отправка данных не происходит
* При успехе 
* При неуспехе 

Для кнопки “Редактировать” добавить ***data-test-id=’edit-button’***, для кнопки “Сохранить изменения” добавить ***data-test-id=’save-button’.*** Для обертки формы добавить ***data-test-id=’profile-form’***

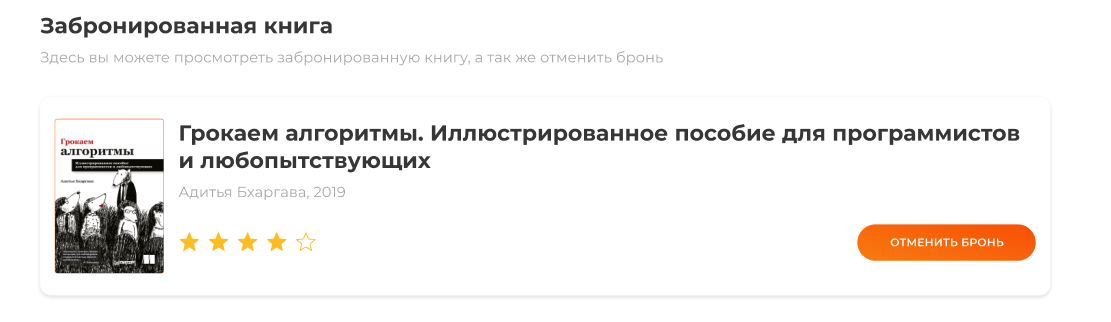
1. **Манипуляции с книгами**

Если никаких данных о книгах у пользователя нет поля ***booking, delivery, history*** - отображаем соответствующую информацию.



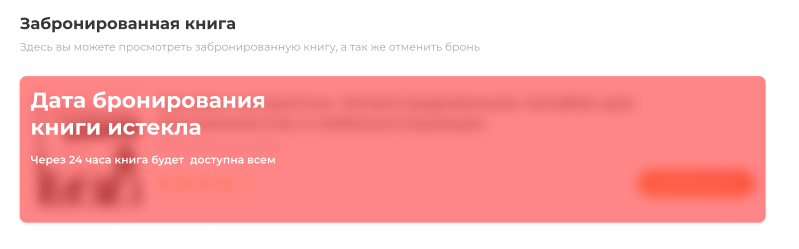
Для пустого окна с синей заливкой добавить ***data-test-id=’empty-blue-card’***

Когда в ***booking*** приходят данные о заказанной книге - отображаем книгу в формате ***list***

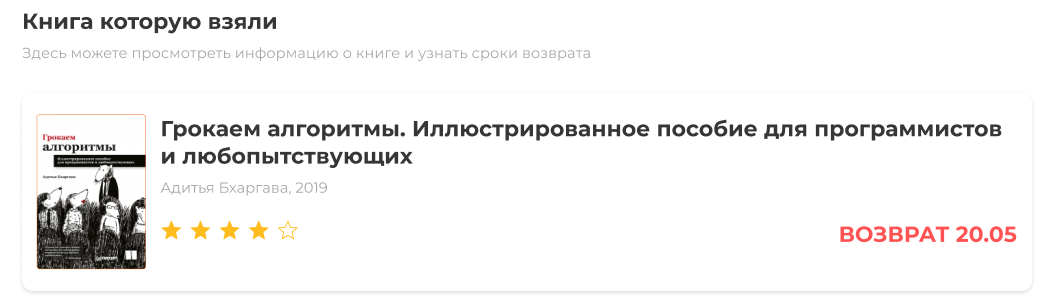
****

Кнопка “Отменить бронь” - завершает бронирование книги с соответствующими сообщениями об успехе или ошибке.

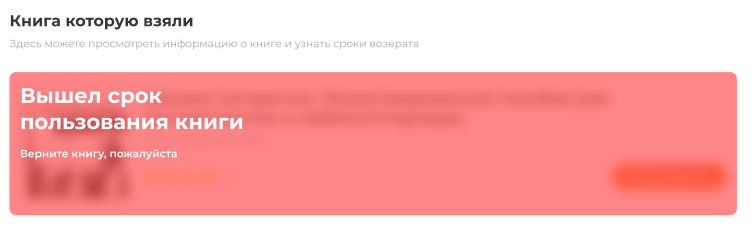
Если книга была забронирована и бронирование просрочено отображается красная маска



Когда в ***delivery*** приходят данные о книге которая сейчас на руках - отображаем книгу в формате ***list*** с датой возврата



Если книга на руках и просрочена дата возврата отображается красная маска



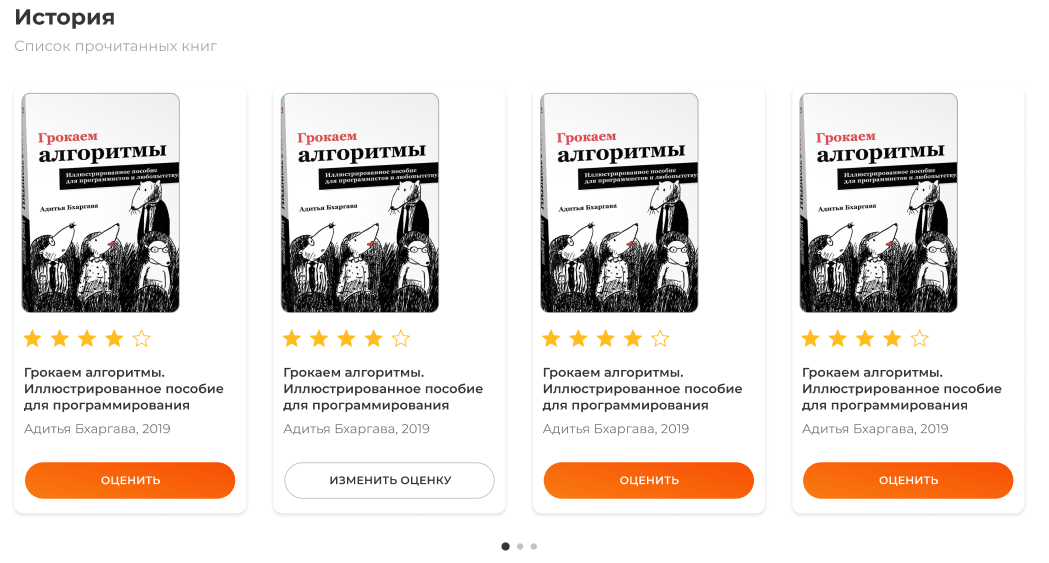
Кнопке “Отменить бронь” добавить ***data-test-id='cancel-booking-button'***

Красной маске добавить ***data-test-id=’expired’***

Когда в ***history*** приходят данные о книгах которые пользователь уже читал - отображаем книги в формате ***window*** в слайдере. При этом текст кнопки меняется на “Оценить” если пользователь еще не оценил книгу или “Изменить оценку” если пользователь уже оставлял оценку с отзывом

Обертке истории добавить ***data-test-id='history'***

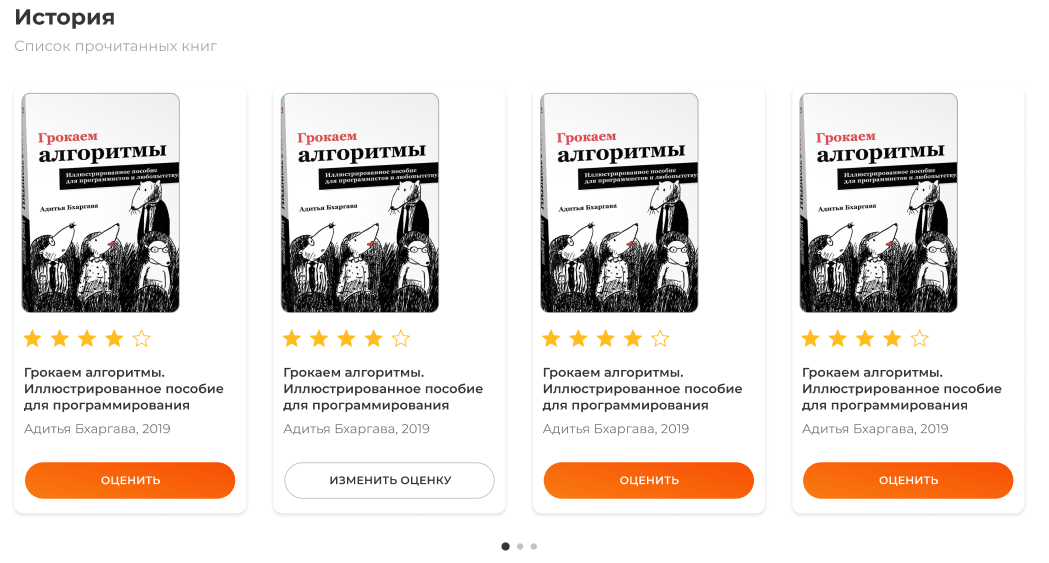
Кнопке в карточке добавить ***data-test-id='history-review-button'***

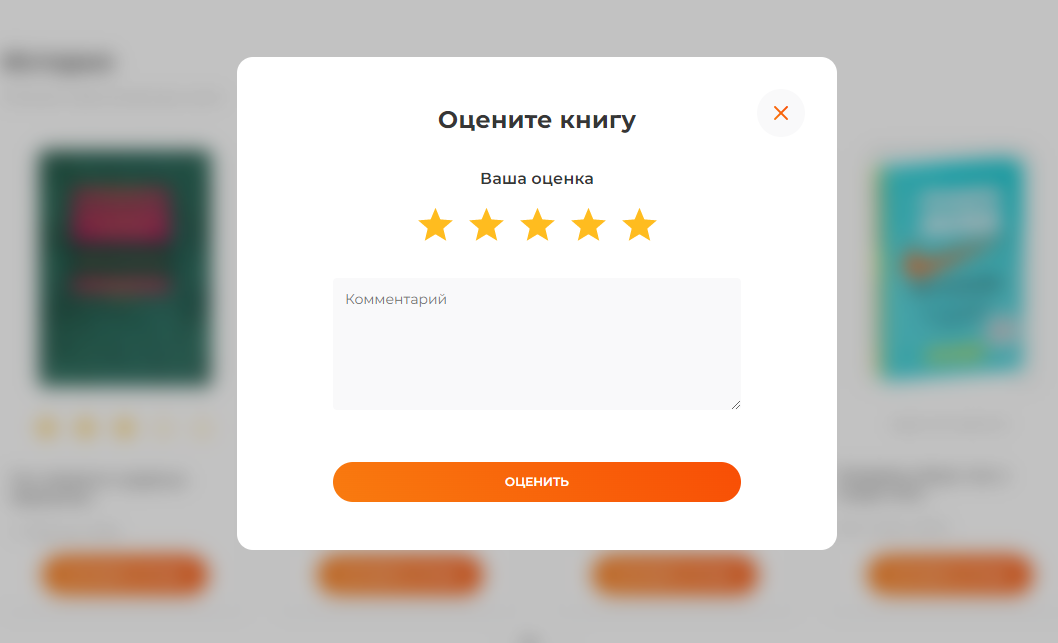
Каждому слайду в истории добавить ***data-test-id='history-slide'***

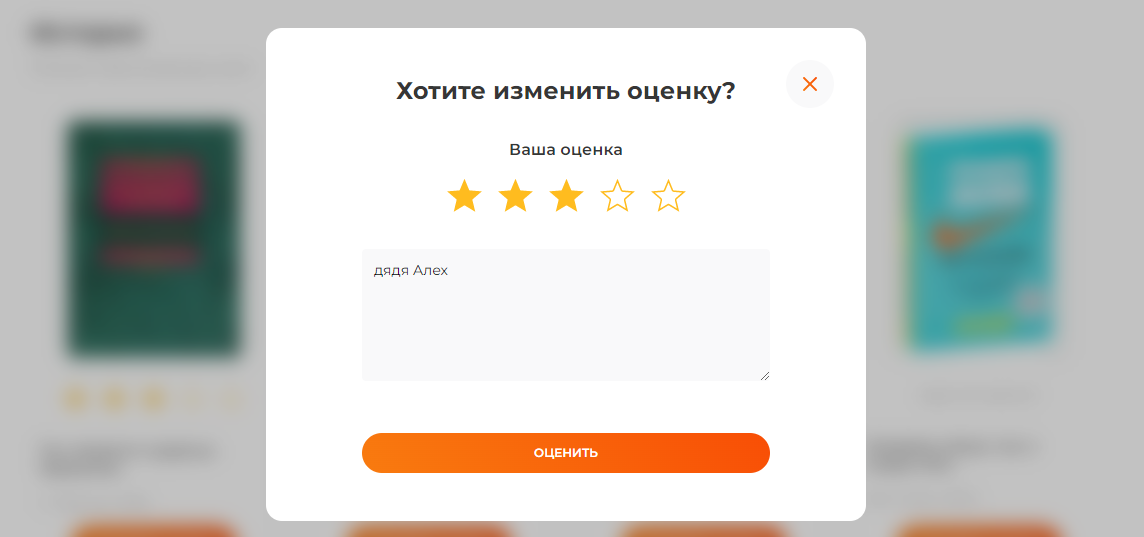
Редактирование комментариев

URL: /api/comments/{commentId}  
метод: PUT

[Подробнее API](https://docs.google.com/document/d/1B5DmTgpTDSlyUpWu4EbyHxi_hyccoBSP4ifX6Qr3LOQ/edit)



При нажатии на кнопку “Оценить” происходит открытие модального окна оценки книги - по умолчанию в ***textarea*** пусто - ***rating*** максимальный - перехода на страницу книги не происходит

При нажатии на кнопку “Изменить оценку” происходит открытие модального окна оценки книги - по умолчанию в ***textarea*** последний оставленный комментарий - ***rating*** рейтинг проставленный для комментария - перехода на страницу книги не происходит

При успехе - тост сообщение об успехе, при ошибке - тост сообщение об ошибке.

**Итого**

Не забываем про адаптивную верстку! Всё должно быть идеально!  
**Дедлайн: 27.03.2023**

Удачи!

P.S.: В файл ***cypress.config.ts*** добавьте в ***e2e*** строчку ***experimentalSessionAndOrigin: true.*** Должно получиться вот так.



Не теряем ***data-test-id*** из предыдущих спринтов - **будут прогоняться все тесты из предыдущих спринтов!**