МШП 20.35 УНИВЕРСИТЕТ КОД БУДУЩЕГО Минцифры России НАЦИОНАЛЬНЫЕ ПРОЕКТЫ РОССИИ ЦИФРОВАЯ ЭКОНОМИКА

Fullstack-разработка

NodeJS
Формы
POST-запросы

МШП

Цель урока:

Научимся отправлять данные **безопасно**

МШП

Разбор ДЗ

МШП

Практика

Заметки

МШП

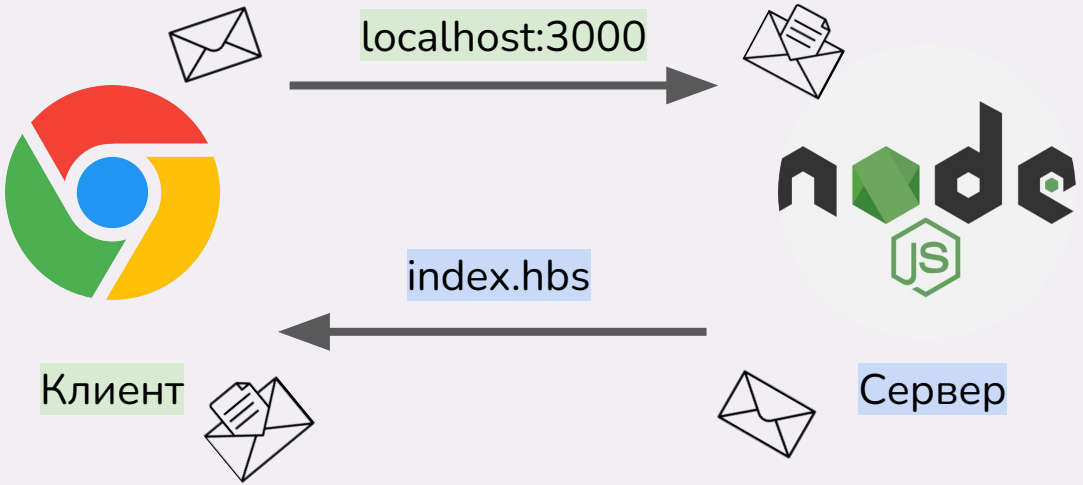
Теория

Виды запросов

МШП

Клиент и сервер взаимодействуют между собой через **запрос** и **ответ.**

Весь процесс похож на доставку писем в конвертах.



МШП

Передача данных происходит через протокол **HTTP**, который задаёт правила написания "писем".

Самое главное правила письма:
Конверт должен быть подписан!

МШП

Примеры

На "конверте" должно быть написано:

1. Куда его доставить
2. Каким способом его доставить

```
GET / HTTP/1.1
```

```
GET /show?id=1 HTTP/1.1
```

```
GET /login?username=Тестер&password=1234
HTTP/1.1
```

МШП

Хьюстон, у нас проблема...

Пользователь указал секретную информацию!

```
GET /login?username=Тестер&password=1234
HTTP/1.1
```

Конверт видят все сотрудники нашей сетевой "почты":

1. **Браузер** сохраняет ссылку в историю посещений
2. **Сервер** занесёт эти данные в лог
3. **Человек**, стоящий рядом, увидит ссылку в строке браузера

МШП

Решение

Пересылать секретные данные **внутри конверта:**

```
POST /login HTTP/1.1
...
username: Тестер
password: 1234
```

1. Использовать вид запроса **POST**,
   а не GET
2. Передавать данные в **теле HTTP запроса**,
   а не через query-параметры

Хорошая новость: поменять GET-запрос на POST-запрос очень-очень просто

МШП

Теория

POST-запрос

МШП

Сервер

Для начала нужно включить в нашем сервере обработку **тела HTTP запроса:**

```
app.use(express.urlencoded({ extended: true }))
```

Размести эту строчку там же, где находятся сейчас другие вызовы метода **app.use,** то есть в начале файла **index.js.**

МШП

Отправка данных

По умолчанию форма делает GET-запросы.

Но эту настройку можно изменить через атрибут **method:**

```
<form action="/login" method="POST">
    <input type="text" name="username">
    <input type="password" name="password">
    <button type="submit">
        Отправить
    </button>
</form>
```

МШП

Обработка данных

app.get → app.post

res.query → res.body

```
app.post(`/login`, function (res, req) {
   let username = res.body.username;
   ...
})
```

МШП

Пример

```
// Рендер шаблона с формой
app.get(`/login`, function (res, req) {
   req.render(`login`);
})

// Обработка запроса
app.post(`/login`, function (res, req) {
   let username = res.body.username;
   res.render(`welcome`, {
       username: username
   });
})
```

МШП

GET-запрос

1. Для рендера шаблонов
2. Для поиска или фильтрации
3. Для действий, не требующих секретности

МШП

POST-запрос

1. Для форм с секретными данными
2. Для форм с персональными данными
3. Для больших объёмов информации

МШП

Теория

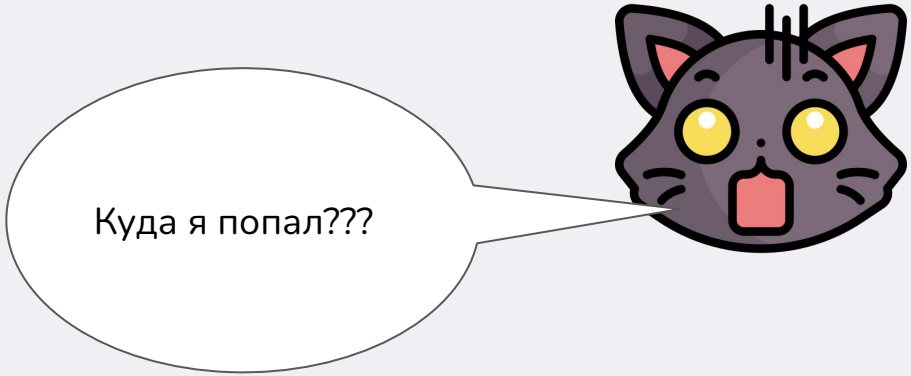
Редирект

Не имеет значения, как пользователь отправляет данные.

Сейчас **он может легко заметить это по изменению данных в URL**

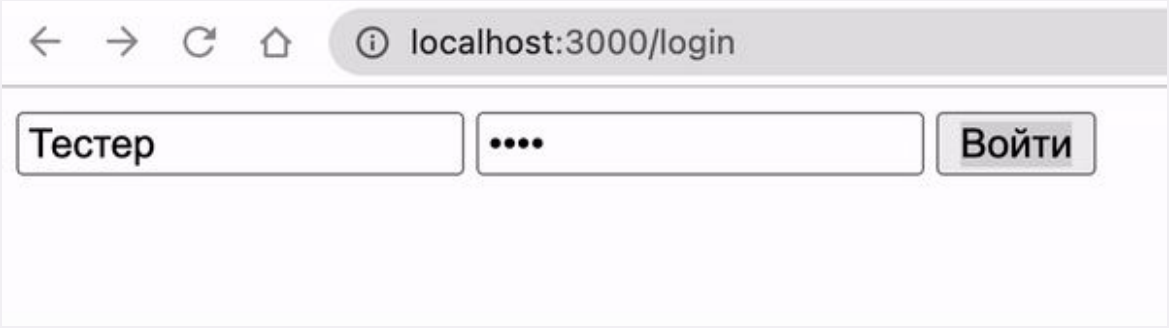
МШП

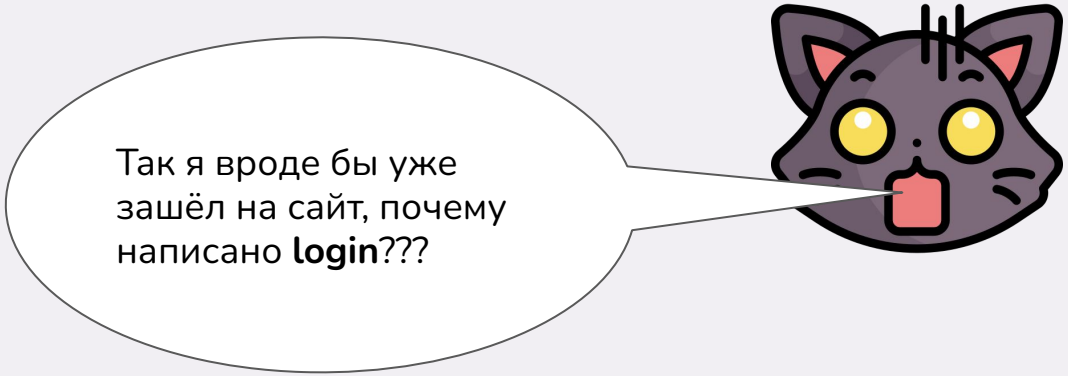
GET-запрос



МШП

POST-запрос





МШП

Редирект – это перенаправление клиента на другой маршрут **незаметно для пользователя**

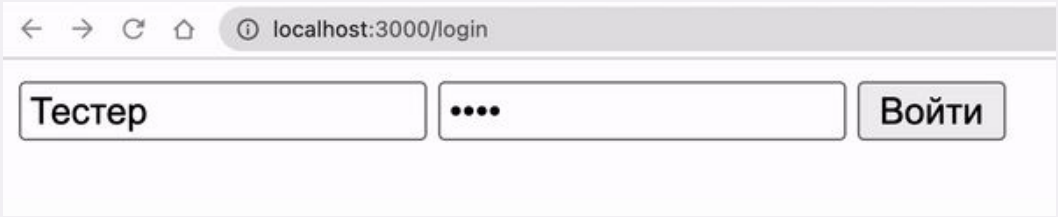
МШП

Пример

```
// Главная страница
app.get(`/welcome`, function (res, req) {
   req.render(`welcome`);
})
// Рендер шаблона с формой
app.get(`/login`, function (res, req) {
   req.render(`login`);
})
// Обработка запроса
app.post(`/login`, function (res, req) {
  // после всех условий и проверок:
   res.redirect(`/welcome`);
})
```

МШП

Пример




```
res.redirect(`/welcome`);
```

Используйте редирект, чтобы показать пользователю нужную страницу после отправки формы.

МШП

Итоги

На практике нам пригодятся:

1. POST-запрос для безопасного добавления заметки
2. GET-запрос для удаления заметки через нажатие на ссылку
3. redirect, чтобы пользователь ничего не заметил :)

МШП

Практика

Доработаем заметки

МШП

Цель урока:

✓ Научились отправлять данные безопасно

МШП

Домашнее задание