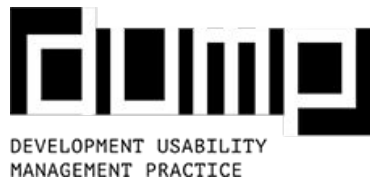


# IT-Practicum

ДЛЯ ТЕСТИРОВЩИКОВ

# Познакомимся



# ЧЕМУ ХОТИМ НАУЧИТЬСЯ?

- Находить дефекты
- Описывать дефекты

# ЧЕМУ ХОТИМ НАУЧИТЬСЯ?

- Планировать действия
- Находить дефекты
- Описывать дефекты

# ЧЕМУ ХОТИМ НАУЧИТЬСЯ?

- Планировать действия
- Находить дефекты
- Описывать дефекты
- **Оценивать ситуацию**

# ЧЕМУ ХОТИМ НАУЧИТЬ?

- Планировать действия
- Находить дефекты
- Описывать дефекты
- Оценивать ситуацию
- **Взаимодействовать с командой**

# ЧЕМУ ХОТИМ НАУЧИТЬСЯ?

- Планировать действия
- Находить дефекты
- Описывать дефекты
- Оценивать ситуацию
- Взаимодействовать с командой
- **Быть успешным тестировщиком**



# НО!

- Приложения разные
- Дефекты разные
- Баг-трекеры разные
- Отчеты разные
- Ситуации разные
- Команды разные



# НО!

- Приложения разные
- Дефекты разные
- Баг-трекеры разные
- Отчеты разные
- Ситуации разные
- Команды разные
- Цели разные

**НО!**

Всё не перепробуем

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

- Собрать информацию о продукте
  - документация
  - заинтересованные лица
  - решаемые проблемы
  - артефакты тестирования
  - сроки тестирования
  - сроки исправления
  - ресурсы
  - ...

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

- Поставить цели тестирования
  - исследовать приложение
  - проверить новую функциональность\*
  - проверить старый функционал\*
  - потратить бюджет
  - сделать отчётик для клиента
  - ...

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

- Ставим приоритеты

- используем *модные* термины

- не вникаем в детали

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

- «Подгоняем ответ»
  - экспертная оценка  
(год назад делали похожее и сломалось)
  - частые баги  
(оно всегда здесь падает)
  - детали реализации  
(чинил Вася)

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

Очевидно же!

В чем проблема?

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

## Ошибка #1

Оперируем дефектами,  
а не функциональностью



# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

Последствия?

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

Последствия

**Надо проверить всё,  
дайте нам месяц.**

# ПЛАНИРОВАТЬ ДЕЙСТВИЯ

## Ошибка #2

Тестируем сначала простые  
вещи

# НАХОДИТЬ ДЕФЕКТЫ В СРОК

- техники и методики поиска\*
- типичные дефекты приложений
- специфичные дефекты приложений
- влияние команды
- расставлять приоритеты

# ОПИСЫВАТЬ ДЕФЕКТЫ ХОРОШО

- краткое описание
- описание по шагам
  1. Нажать «Удалить»
  2. Ввести в поле кол-во «666»
- реальный результат
- ожидаемый результат
- лог, скриншот, мысли в слух

# ОПИСЫВАТЬ ДЕФЕКТЫ ХОРОШО

Не боимся писать очевидные вещи

- через неделю все поменяется
- у разработчика другое любимое число
- разработчик не знает приложение\*

# ОЦЕНИВАТЬ СИТУАЦИЮ АДЕКВАТНО

- критичность и приоритет
- много дефектов – это сколько?
- когда прекращать тестировать?
- кто виноват?
- что делать? 😊

# ОЦЕНИВАТЬ СИТУАЦИЮ АДЕКВАТНО

- критичность и приоритет
- много дефектов – это сколько?
- когда прекращать тестировать?
- кто виноват?
- что делать? 😊



# ОЦЕНИВАТЬ СИТУАЦИЮ АДЕКВАТНО

- критичность и приоритет
- много дефектов – это сколько?
- когда прекращать тестировать?
- кто виноват?
- что делать? 😊

# ОЦЕНИВАТЬ СИТУАЦИЮ АДЕКВАТНО

- критичность и приоритет
- много дефектов – это сколько?
- когда прекращать тестировать?
- кто виноват?
- что делать? 😊

# ОЦЕНИВАТЬ СИТУАЦИЮ АДЕКВАТНО

- критичность и приоритет
- много дефектов – это сколько?
- когда прекращать тестировать?
- **КТО ВИНОВАТ?**
- что делать? 😊

# ОЦЕНИВАТЬ СИТУАЦИЮ АДЕКВАТНО

- критичность и приоритет
- много дефектов – это сколько?
- когда прекращать тестировать?
- кто виноват?
- что делать? 😊

# ВЗАИМОДЕЙСТВИЕ С КОМАНДОЙ

Тестировщик вне команды –  
обезьянка нажимающая  
кнопочки



# **ОСОБЕННОСТИ ТЕСТИРОВАНИЯ WEB-ПРИЛОЖЕНИЙ**

# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

Страничка в браузере

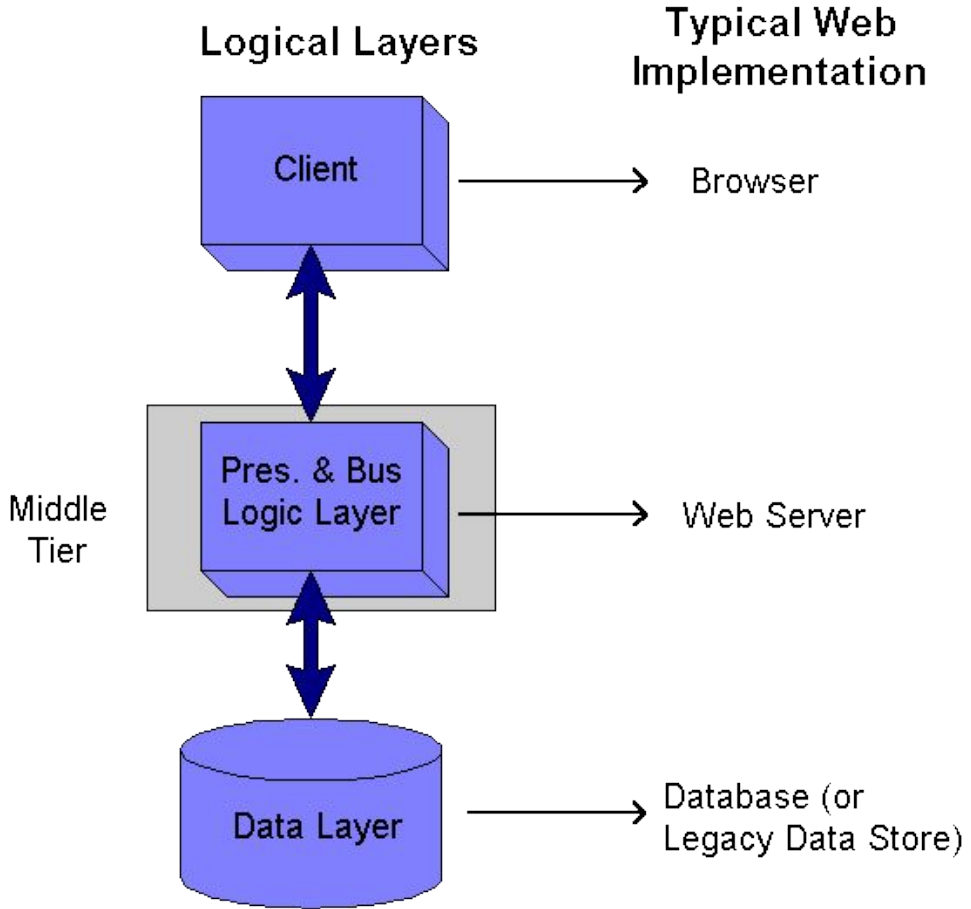
Что еще?

# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

- сервер приложений
- сервер баз данных



# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ



# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

- сервер приложений
- сервер баз данных

# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

- сервер приложений
- сервер баз данных
- проксирующий сервер

# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

- сервер приложений
- сервер баз данных
- проксирующий сервер
- кеширующий сервер

# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

- сервер приложений
- сервер баз данных
- проксирующий сервер
- кеширующий сервер
- failover кластер

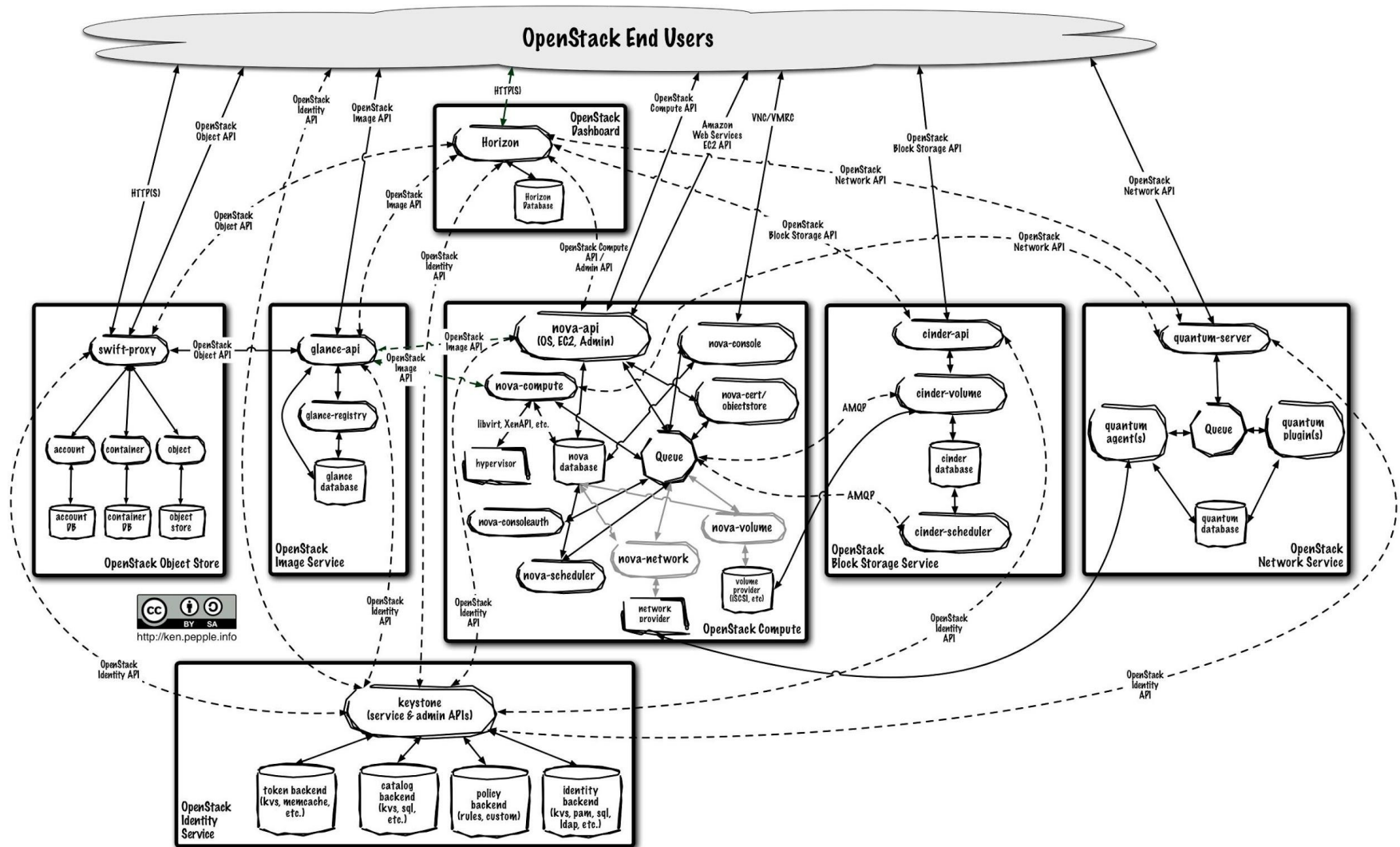
# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

- сервер приложений
- сервер баз данных
- проксирующий сервер
- кеширующий сервер
- failover кластер
- CDN

# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ

- сервер приложений
- сервер баз данных
- проксирующий сервер
- кеширующий сервер
- failover кластер
- CDN
- ...

# АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЙ





# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Факторы риска

- фрагментированность браузеров
- динамический контент (CRUD)
- динамический интерфейс (AJAX)
- сессии/авторизация
- инфраструктура сети
- интеграции

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Фрагментированность браузеров

- поддерживаемые версии
- VirtualBox (облачные сервисы)
- известные ошибки

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Фрагментированность браузеров

- поддерживаемые версии
- VirtualBox (облачные сервисы)
- известные ошибки

**в IE6 много «неожиданностей», но все  
уже знают как их обходить**

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Фрагментированность браузеров

- поддерживаемые версии
- VirtualBox (облачные сервисы)
- известные ошибки

**В Chrome много новых «плюшек» и надо постоянно что-то придумывать**

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Фрагментированность браузеров

- поддерживаемые версии
- VirtualBox (облачные сервисы)
- известные ошибки  
Firefox «особенностей реализации»

~~В Chrome много новых «плюшек»~~ и надо  
постоянно что-то придумывать

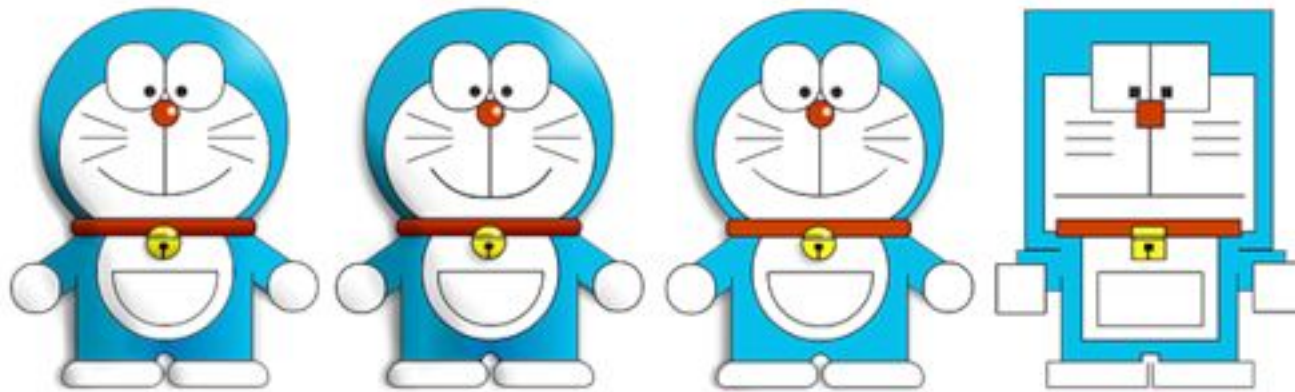
# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Фрагментированность браузеров

- поддерживаемые версии
- VirtualBox (облачные сервисы)
- известные ошибки
- graceful degradation

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Graceful Degradation



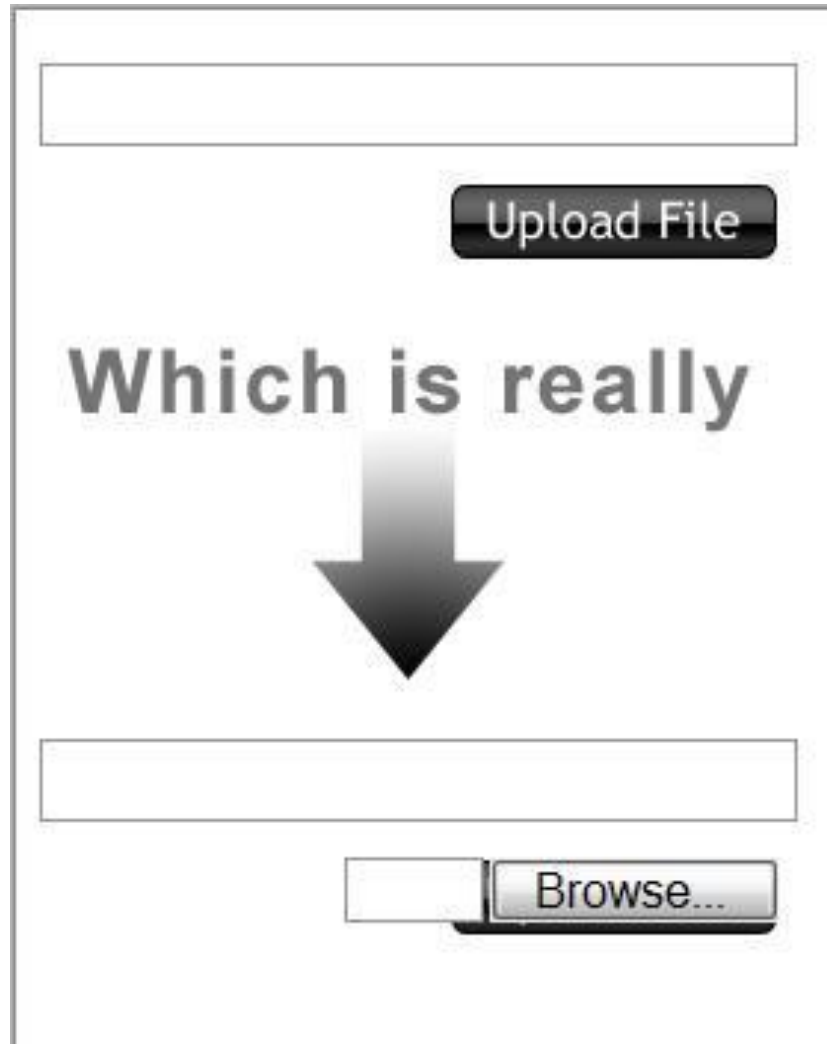
Chrome 5.0

Firefox 3.6

Opera 10.53

IE 8

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ





# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Динамический контент

- Create
- Read
- Update
- Delete

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Динамический контент

- |          |        |           |
|----------|--------|-----------|
| • Create | INSERT | POST      |
| • Read   | SELECT | GET       |
| • Update | UPDATE | PUT/PATCH |
| • Delete | DELETE | DELETE    |

**DB**

**HTTP**

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Задание

<http://listmoz.com/>

Написать список проверок на каждый из видов работы с динамическим контентом (Create, Read, Update, Delete) и по 2 тест-кейса

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Динамический интерфейс/SRA

- скрытый обмен данными
- сложный интерфейс
- трудности с навигацией

# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Steps



Steps	Action	Expected result
5.	Click on the user list. Click on the name of the user in the list you populated.	Should display the profile of the selected user.

# **ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ**

## **Консоль разработчика**

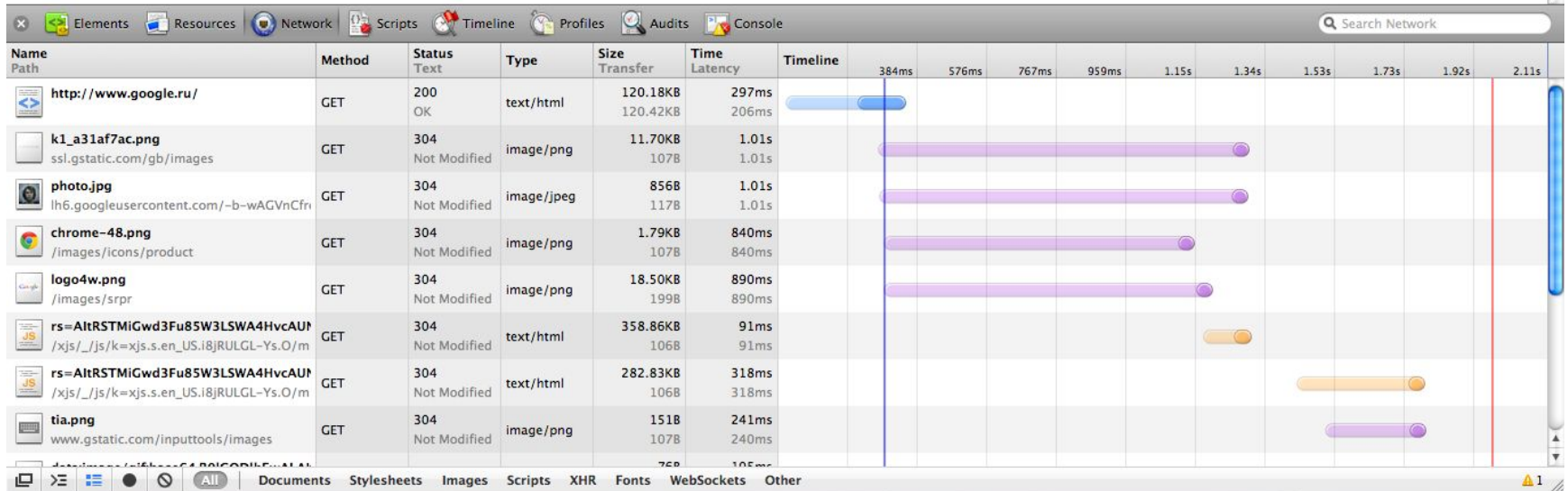
# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Консоль разработчика



Поиск в Google

Мне повезёт!



# ФУНКЦИОНАЛЬНОЕ

The image shows a web browser window with a login form overlay. The form has the following elements:

- Text: "All fields with \* are required"
- Error message: "The user with the provided email address is not recognized."
- Input field: "\*Email address:" with the value "test@mail.com"
- Input field: "\*Password:" with masked characters "....."
- Button: "Sign In"
- Link: "Forgot password? [Click here](#)"

The browser's developer tools are open, showing the Network tab with a request to "login.login.form.json". The response is a JSON object:

```
1 {"errorMessage":"cap user does not exist.,"errorCode":2198,"success":false}
```

The status bar at the bottom indicates "1 requests | 310B transferred".



# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

## Консоль разработчика



# ФУНКЦИОНАЛЬНОЕ ТЕСТИРОВАНИЕ

<https://httpstat.us>

# ДОМАШНЕЕ ЗАДАНИЕ

## Задание #1

<http://listmoz.com/>

Написать список проверок на каждый из видов работы с динамическим контентом (Create, Read, Update, Delete) и по 2 тест-кейса

## Задание #2

1. Выбрать сайт
2. Составить план для smoke-тестирования

# ССЫЛКИ

- **Magic Numbers**

<http://msdn.microsoft.com/en-us/library/ee621251.aspx>