**LEVEL1**

1.

|  |  |
| --- | --- |
| Роль | Хорошая верстка |
| Пользователь | 1. Хороший, чистый, дизайн.  2. Понятная система навигации по сайту. 3. Приятные анимации. 4. Отображение результатов действий (нажатие, отправление, удаление, отмена). 5.Сайт/приложение/интерфейс должны корректно отображаться во всех браузерах и на всех устройствах. |
| Менеджер проекта | 1.Выполнение всех пунктов и требований ТЗ 2. Хорошо структурированный репозиторий с кодом. |
| Дизайнер | 1. Продуманная цветовая схема 2. Анимации 3. Продуманные размеры текста. |
| Верстальщик | 1. Соответствие макету. 2. Pixel Perfect. 3. Адаптивность. 4. Сохранение пропорций элементов. |
| Клиентский программист | 1. Хорошо структурированный код, 2. Продуманные элементы, которые могут быть использованы снова. |
| Серверный программист | 1. Присутствие элементов, которые отображают все данные, полученные с сервера.   2. Наличие форм, позволяющих делать верные запросы на сервер, которые будут проходить валидацию.  3.Кэширование запросов, server-side rendering |

2.

Основной особенностью верстки многостраничных сайтов является то, что необходимо с самого начала продумать, из каких элементов будет состоять каждая страница, затем написать максимально переиспользуемые компоненты, которые будет удобно изменять от странице к странице.

При верстке таких сайтов лично я люблю использовать следующий набор инструментов

1. NextJS и React – в комбинации позволяют создавать веб-приложения с улучшенной производительностью благодаря SSR, SSG. Удобно создавать страницы, удобный сборщик проекта.
2. Redux – позволяет управлять состоянием элементов, убирает нужду в prop drilling.
3. Tailwind, SASS, React Styled Components - любой из этих способов написания стилей

3.

mobile-first верстка – потому что как трафик с мобильных устройств возрастает.

Первоначально верстка тестируется в браузере, с помощью инструментов разработчика меняю тип устройства и проверяю корректность работы. Затем обязательное тестирование на живых устройствах – потому как тот же Chrome для различных девайсов работает иногда по-разному.

4.

Панель разработчика Google-chrome.

React, TypeScript, ESLint, prettier. Разнообразные плагины в VS code (сниппеты)

5.

Можно открыть инструменты разработчика, которые позволят быстро посмотреть на проблему. Если проблема в CSS, нужно дописать стили для каждого проблемного браузера. Можно так же закинуть сайт на Markup Validation Service, с помощью него можно выявить устранить множество ошибок, относящихся к отображения на разных браузерах.

6.

В первую очередь, нужно уточнить у дизайнера, является ли это финальным вариантом макета. Если не будет обратной связи, то сделаю свой стиль для наведения, который потом можно будет быстро изменить или убрать, после уточнения или перед запуском в продакшн

7.

Для практики JS и алгоритмов – пользуюсь LeetCode и CodeWars.

Для изучения новой технологии пользуюсь YouTube – Каналы Web Dev Simplified, Pedro Tech и в целом любой канал, имеющий информацию, которая мне нужна на данный момент. RocketDevBlog и такие как CSS- tricks, habr, HTML-academy.

Мне более близко и интересно JS программирование и фреймворки основанные на нем.

Области знаний, которые мне близки и интересны – иностранные языки.

8.

**Немного о себе:**  
A) Высшее образование по направлению Прикладная Информатика.

Б) Способен выучить все что угодно, это необходимо (в университете не было Frontend разработки, все что я выучил - я выучил и разобрал самостоятельно).

Больше информации вы можете на либо, в портфолио либо в резюме. Ссылки ниже.

GitHub - <https://github.com/Daniellios>

Ссылка на резюме - <https://hh.ru/resume/92b2ea36ff09bd84230039ed1f6f644f304b39>

Ссылка на портфолио - <https://my-portfolio-blue-pi.vercel.app/>