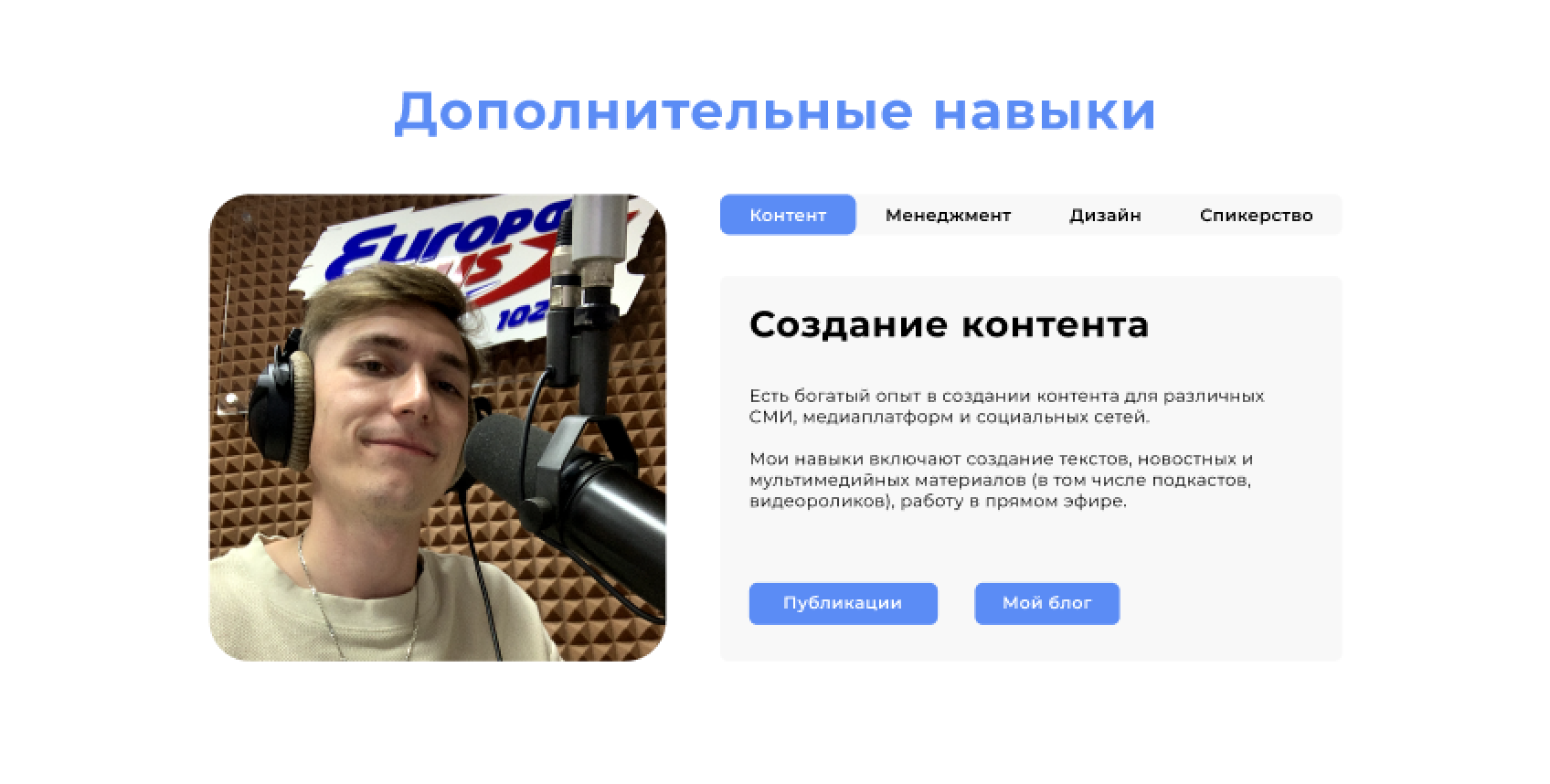
**Сайт-визитка**

**Блок «Дополнительные навыки» / «Опыт»**

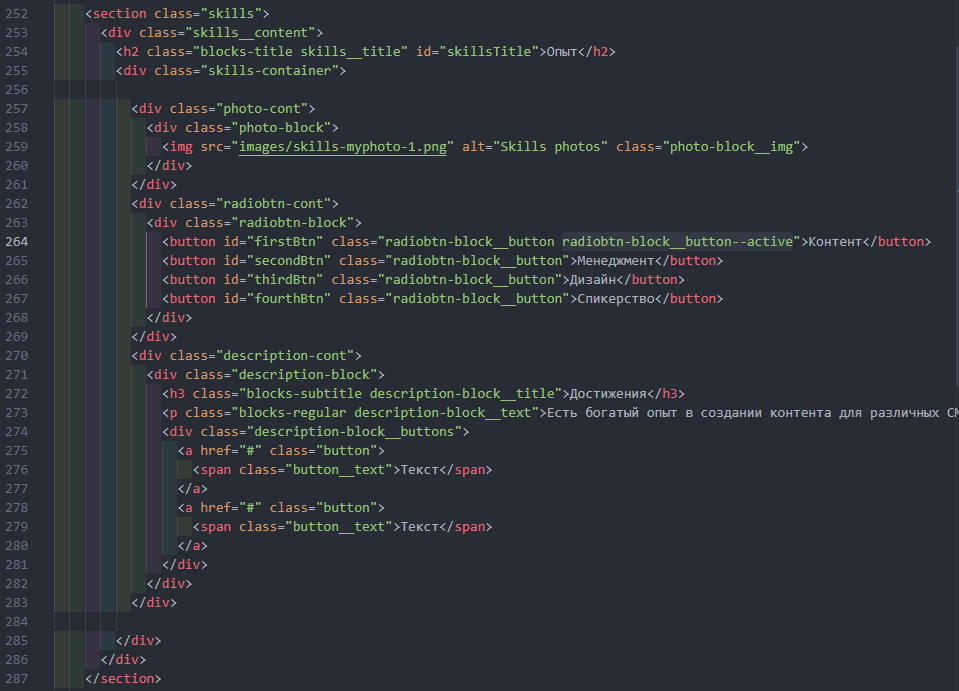


В контексте сайта был назван «skills».

Нужно реализовать:

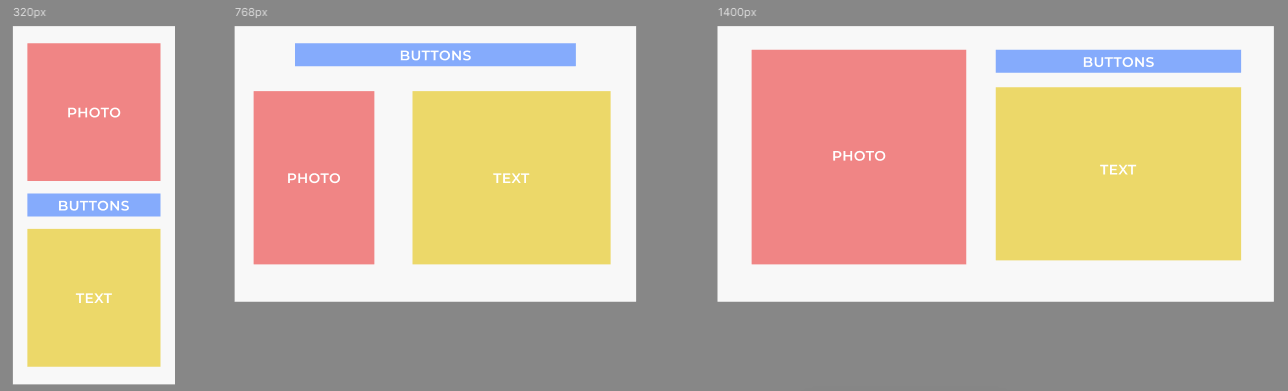
1. Заголовок
2. Блок с контентом:
   1. Блок с фотографией
   2. Блок с кнопками (пагинацию)
   3. Блок с текстом и ссылками

В итоге HTML-структура представилась таким образом:

Для реализации блока пагинации пришлось убрать глобальный блок .container, который задавал padding по бокам секции.

Однако в других блоках эти отступы нужны, поэтому было принято решение поместить каждый в доп. блок -cont, с помощью которого был задан свой отступ.

При этом все блоки внутри должны менять свою позицию в зависимости от разрешения экрана:



Для решения данной проблемы нужно было применить grid + свойство grid-template-areas:

  .skills-container {

*// max-width: 320px;*

*// width: 100%;*

    overflow: hidden;

    justify-content: center;

    display: grid;

    gap: 20px;

    grid-template-areas:

        "a a a"

        "b b b"

        "c c c";

        @media (min-width: 768px) {

          gap: 25px;

          grid-template-areas:

          "b b b"

          "a c c"

          "a c c";

        }

        @media (min-width: 1400px) {

          gap: 50px;

          padding-top: 30px;

          grid-template-areas:

          "a a b"

          "a a c"

          "a a c";

        }

Это позволило размещать все блоки также, как и в макете. Единственный недостаток: размеры некоторых блоков пришлось задать точно, чтобы не было выходов за пределы.

Функциональность:

Пагинация должна позволять просматривать разное содержимое с помощью кнопок «Контент», «Менеджмент», «Дизайн» и «Спикерство».

Нажатие на одну из кнопок добавляет ей стиль и меняет текстовое содержимое и фотографию на соответствующую названию. По умолчанию активной считается «Контент».

Реализация замены содержимого после нажатия:

Сначала я создал массив данных, куда занёс все значения, которые будут отображаться в блоке. Это изображение, заголовок, текст-описание и кнопки:

const data = {

  content: {

    image: "images/skills-myphoto-1.png",

    title: "Создание контента",

    description: "Есть богатый опыт в создании контента для различных СМИ, медиаплатформ и социальных сетей. <br><br>Мои навыки включают создание текстов, новостных и мультимедийных материалов (в том числе подкастов, видеороликов), работу в прямом эфире.",

    buttons: [

      {

          text: "Публикации",

          link: "https://docs.google.com/document/d/1vXn8nLEUYs1dHJ1VDHtr8BRPOlq6yE31/edit?usp=drive\_link&ouid=105516434251710796534&rtpof=true&sd=true",

      },

      {

          text: "Мой блог",

          link: "https://t.me/prolisaiezha",

      },

    ]

  },

  management: {

    image: "images/skills-myphoto-2.jpg",

    title: "Управление проектами",

    description: "В рамках учебных дисциплин разрабатывал личный проект и был лидером группового проекта.<br><br>Разработал идею стартапа «YOZH» и привёл команду к победе в 3 конкурсах грантов и других соревнованиях. Стал основателем компании ООО «ИТС».",

    buttons: [

      {

          text: "Стартап",

          link: "https://app.yozh.fun",

      }

    ]

  },

  design: {

    image: "images/skills-myphoto-3.png",

    title: "Работа с графикой",

    description: "Прошёл несколько курсов по дизайну и графике, имею практический опыт работы с Illustrator, Photoshop, InDesign, Figma, создания презентаций, а также нейросетями для генерации изображений.<br><br>Провёл ребрендинг одной из медиаструктур вуза, активно работал с вёрсткой бумажных материалов.",

    buttons: [

      {

          text: "Работы",

          link: "https://drive.google.com/drive/folders/1-38m2VxYn5LVjk0S10IFXenghpaATLcl?usp=drive\_link",

      }

    ]

  },

  speaker: {

    image: "images/skills-myphoto-4.png",

    title: "Публичные выступления",

    description: "Выступил в роли спикера на +50 мероприятиях (презентации и защиты личного проекта, мастерклассы, самопрезентации) в разных городах страны.<br><br> Большой опыт общения со СМИ (интервью, репортажи, передача информации о проекте)",

    buttons: [

      {

          text: "Материалы со мной",

          link: "https://docs.google.com/document/d/1f9vpoEaMH3tTZOac3JzJizSu0lZFx5ozsLxKdTyOFkw/edit?usp=drive\_link",

      }

    ]

  }

};

Чтобы обращаться к разным кнопкам, были созданы отдельные объекты: content, management, design и speaker.

Далее была реализована функция для обращения к кнопкам и определения их id. Определение id позволит понять, из какого объекта в массиве мы будем вытаскивать данные:

const buttons = document.querySelectorAll('.radiobtn-block\_\_button');

buttons.forEach(button => {

  button.addEventListener('click', function(event) {

    buttons.forEach(btn => {

      if (btn !== this) {

        btn.classList.remove("radiobtn-block\_\_button--active");

      }

    });

*// Добавляем класс только к нажатой кнопке, если он еще не добавлен*

    this.classList.add("radiobtn-block\_\_button--active");

    if (event.target.id === 'content') {

      renderData('content');

    } else if (event.target.id === 'management') {

      renderData('management');

    } else if (event.target.id === 'design') {

      renderData('design');

    } else if (event.target.id === 'speaker') {

      renderData('speaker');

    }

  })})

Логика кода:

1. Определяем все кнопки с классом **.radiobtn-block\_\_button**;
2. Перебираем кнопки и к каждой применяем функцию, которая удаляет модификатор **--active**;  
   *Вроде бы тут нет связи с обработчиком событий, как и с отдельным элементом button из массива. Но на деле эта часть кода должна запускаться каждый раз, когда мы нажимаем на кнопку, чтобы снять модификатор со всех кнопок. А после этого следует присваивание отдельной кнопке класса. Поэтому этот код тут.*
3. Добавляем модификатор нажатой кнопке;
4. Определяем, какой id у нажатой кнопки, чтобы обратиться к её объекту в массиве;
5. Запускаем функцию с аргументом в виде id нажатой кнопки.

Функция **renderData**:

  function renderData(blockId) {

    document.getElementById('skillsImg').src = data[blockId].image;

    document.getElementById('skillsSubitle').textContent = data[blockId].title;

    document.getElementById('skillsText').innerHTML = data[blockId].description;

    const buttonsHTML = data[blockId].buttons.map(button => {

        return `

            <a href="${button.link}" class="button" target="\_blank">

                <span class="button\_\_text">${button.text}</span>

            </a>

        `;

    }).join(''); *// Объединяем массив строк в одну строку*

    document.getElementById('skillsButtons').innerHTML = buttonsHTML;

}

Логика кода:

1. Принимаем в функции id нашей кнопки как **blockId**;
2. Ищем блок с картинкой и обращаемся к свойству **src**. Заменяем актуальное содержимое свойства на то, которое содержится в объекте кнопки по пути **data[blockId].image**;
3. Таким же образом заменяем текстовое содержимое заголовка;
4. Описание заменяется не как текстовое содержимое, а как всё содержимое блока, т.к. внутри есть теги **<br>** для переноса строк;
5. Для вставки кнопок обращаемся ко всем «кнопкам» (точнее их составляющим) в выбранном объекте и вызываем обработчик на каждую. Этот обработчик возвращает код-конструктор, где динамически вставляется свой текст **${button.text}** и ссылка **${button.link}**. В итоге массив **buttonsHTML** объединяется в одну строку;
6. Вставляем созданные кнопки(у) внутрь блока с id **skillsButtons**;

Также весь заголовок «Дополнительные навыки» не помещался в мобильной версии сайта, поэтому было решено его заменять на «Опыт», если разрешение меньше 768px. Для этого была написана функция, определяющая размеры окна браузера (в т.ч. и при изменении в DevTools):

function updateText() {

  const skillsTitle = document.getElementById('skillsTitle');

  if (window.innerWidth > 768) {

    skillsTitle.textContent = 'Дополнительные навыки';

  } else {

    skillsTitle.textContent = 'Опыт';

  }

}

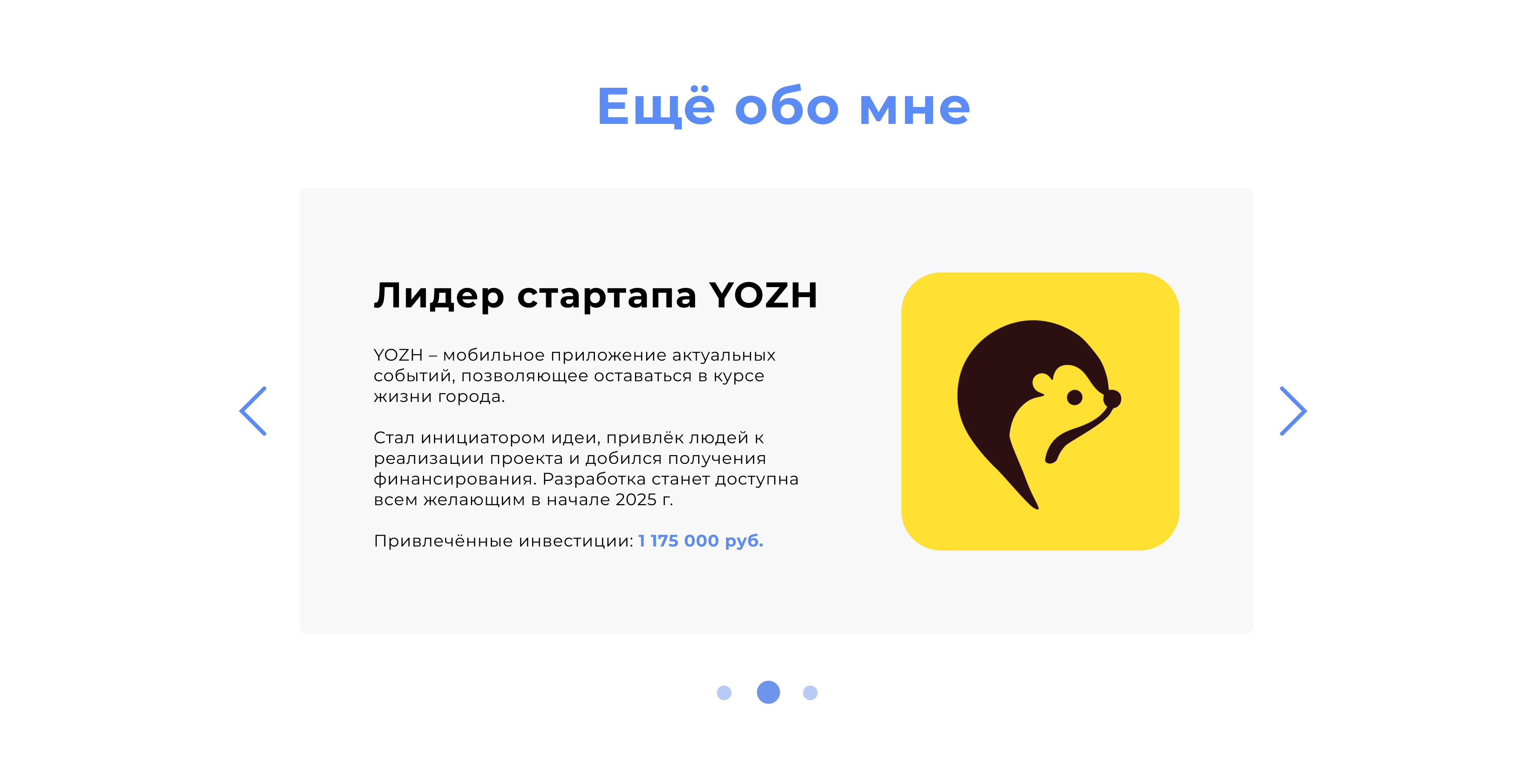
updateText()

window.addEventListener('resize', updateText);

Блок был реализован полностью и без заметных изъянов, все функции работают. Остались некоторые незначительные недостатки / проблемы:

1. Фиксированные значения стилей: сложность подключения резиновой вёрстки. Макет растягивается только до 500px, далее всё меняется по брейкпоинтам. Это оправдывается тем, что блоки трудно адаптировать друг под друга, чтобы содержание не выходило за границы. А также тем, что некоторые блоки невозможно было (с моим опытом) сделать резиновыми. Блок с пагинацией составлен из кнопок, у которых нет точного значения из-за отступов внутри каждой. В зависимости от текста, общая ширина пагинации была разной. Поэтому остальные блоки пришлось подстраивать под неё во всех разрешениях.

**Блок «Ещё обо мне»**



В контексте сайта был назван «bioplus».

Нужно реализовать:

1. Заголовок
2. Свайпер с 3 сладами, которые содержат:
   1. Заголовок слайда
   2. Блок с фотографией
   3. Текст-описание

В итоге HTML-структура представилась таким образом: