Софийски университет „Св. Кл. Охридски”

Факултет по математика и информатика

Специалност
„Софтуерно инженерство”

Курсов проект

XML технологии за семантичен уеб

Зимен семестър, 2023/2024 год.

Тема №16:
„Каталог на ИТ продукти 2“

Автори:

Дейвид Барух ,ФN№ 0MI0600128

Александър Спиридонов ,ФN№ 4MI0600117

Ръководители:

доц. д-р Александър Димов

гл. ас. д-р Явор Данков

януари, 2024

София

Съдържание

1 Въведение (1 стр.[1])

Каталогът от ИТ-продукти предлага разнообразие от последни модели хардуерни и софтуерни продукти. Хардуерните продукти, които предлага каталога са лаптопи, процесори, видеокарти. Софтуерните продукти, които предлага каталога са антивирусни програми.

Проектът е много актуален във днешно време , защото технологиите много бързо се променят и на човек му трябва най-модерните технологични устройства, които и нашият сайт предлага.

Много от сайтовете не си обновяват продуктите , докато при нас винаги се предлагат съвременни устройства.

Нашата богата гама от продукти е взета от множество сайтове , като предоставяме само най-доброто и съвременно от тях. За тази цел сформираме данните в XML файл, който преобразуваме в HTML страница чрез помощта на XSL файл. Използва се XML Schema за валидиране на данните от XML файла. В XSL файла е добавен javascript код , който спомага за преминаването в различни страници, съдържащи различните продукти. За по-прегледен вид на съдържанието е използвана CSS стилизация.

1.1 Работен процес (2-3 стр.)

Работният процес се състои от следните стъпки:

1. Даните от продуктите се вкарват в XML файл с име itCatalog.xml
2. XML файлът се валидира посредством XML Schema, намираща се в файла itCatalogSchema.xsd.
3. След валидиране на съдържанието, се трансформира itCatalog.xml файлът посредством XSLT скрипт от файла itCatalog.xsl . Като резултат се съствалява .html файл;
4. Генерираният HTML документ реферира mainStyle.css и productsStyle.css - файлове с каскадни стилове за прегледно оформление на резултатното съдържание.

[1] За всяка секция е указан препоръчителен обем в страници

Входното съдържание представлява множество от софтуерни и хардуерни продукти.

Всички продукти споделят следните характеристики :

- Вид - определя продукта като софтуерен или хардуерен. Служи като външен ключ за категоризиране на продукта.
- Цел - определя продукта като предназначен за офис (параметрите на продукта са достатъчно добри за поддържане на работни дейности) или за гейминг (параметрите на продукта са от най-високо ниво, за да може продукта да издържа голямо натоварване) . Служи като външен ключ за категоризиране на продукта.
- Изображение - примерно изображение (entity обект) , което дава представа за визията на продукт.

Хардуерните и софтуерните продукти (лаптопи,процесори,видеокарти,антивирусни програми) . В проектът следните продукти са представени по следния начин :

Лаптоп се характеризира с :

- Име- свободно текстово съдържание, представляващо комерсиалното наименование на лаптоп.
- Дисплей - свободно текстово съдържание, описващо резолюцията на лаптоп.
- Честота на опресняване – целочислено число, което показва колко кадъра в секунда се възпроизвеждат , измерено в Hz(херцове) .
- Процесор - свободно текстово съдържание, което показва модела и името на процесора на лаптопа.
- Видео карта - свободно текстово съдържание, което показва модела и името на видеокартата на лаптопа.
- RAM - целочислено число, което показва обема на RAM паметта в GB (гигабайти).

Примерен обект лаптоп в xml файла :

```
<product iteam="computer" id="comp3">
                <name>ASUS ROG Zephyrus G14 </name>
                <display>Full HD 1920:1080 IPS </display>
                <displayhertzs>144</displayhertzs>
                <processor>AMD Ryzen 9 4900HS Processor </processor>
                <gpu> RTX 2060 Max-Q GPU with Ray Tracing</gpu>
                <ram> 32</ram>
                <type idref="hardware"/>
                <purpose idref="gaming"/>
                <image source ="laptop3_image"/>
            </product>
```

Процесор се характеризира с :

- Име- свободно текстово съдържание, представляващо комерсиалното наименование на процесор.
- Физически ядра - целочислено число, описващо броя на физическите ядра
- Логически ядра - целочислено число, описваща броя на логическите ядра
- Основна честота– дробно число, което показва честота, на която процесора работи при ниско до умерено натоварване.
- Максимална честота– дробно число, което показва честота, на която процесора работи при високо натоварване
- Кеш - целочислено число, което показва големината на кеш паметта в MB (мегабайти).
- Топлинна проектна мощност- целочислено число , което показва нужната за функциониране мощност на процесора измерена във W (ватове).

Примерен обект процесор в xml файла :

```
<product iteam="processor" id="proc3">
            <name>Intel core i9 10900K </name>
            <corephis>10</corephis>
            <corelog>20</corelog>
            <frequencybase>3.7</frequencybase>
            <frequencyboost>5.3</frequencyboost>
            <cache>20</cache>
            <tdp>95</tdp>
            <type idref="hardware"/>
            <purpose idref="gaming"/>
            <image source="processor3_image"/>
        </product>
```

Видеокарта се характеризира с :

- Име- свободно текстово съдържание, представляващо комерсиалното наименование на видеокарта.
- Видеопамет - целочислено число, показващо обема на паметта на видеокартата , измерена в GB (гигабайти).
- Скорост на видеопамет – свободно текстово съдържание, което показва колко бързо се достига паметта.
- Основна честота– дробно число, което показва честота, на която процесора работи при ниско до умерено натоварване.
- Максимална честота– дробно число, което показва честота, на която процесора работи при високо натоварване.

- Bandwidth - целочислено число , представляваща мярка за скоростта на трансфер на данни между видеокарта и системата , измерена в GB/s (гигабайт в секунда).

  Примерен обект видеокарта в xml файла :

```
<product iteam="videocard" id="vid3">
            <name>NVIDIA RTX 3080</name>
            <memory>10</memory>
            <memoryspeed>GDDR6X</memoryspeed>
            <bandwidth>760</bandwidth>
            <frequencybase>1.4</frequencybase>
            <frequencyboost>1.7</frequencyboost>
            <type idref="hardware"/>
            <purpose idref="gaming"/>
            <image source="vidcard3_image"/>
        </product>
```

Антивирус се характеризира с :

- Име- свободно текстово съдържание, представляващо комерсиалното наименование на антивирус.
- Процент на откриване на зловреден софтуер - целочислено число, описващо процентът на успеваемост на антивирусната система да открие зловреден софтуер.
- Брой устройства – целочислено число, което показва колко максимум устройства могат да се възползват от антивирусната система под един акаунт.
- Съвместими операционни системи - свободно текстово съдържание, което показва операционни системи , които поддържат антивирусната система.

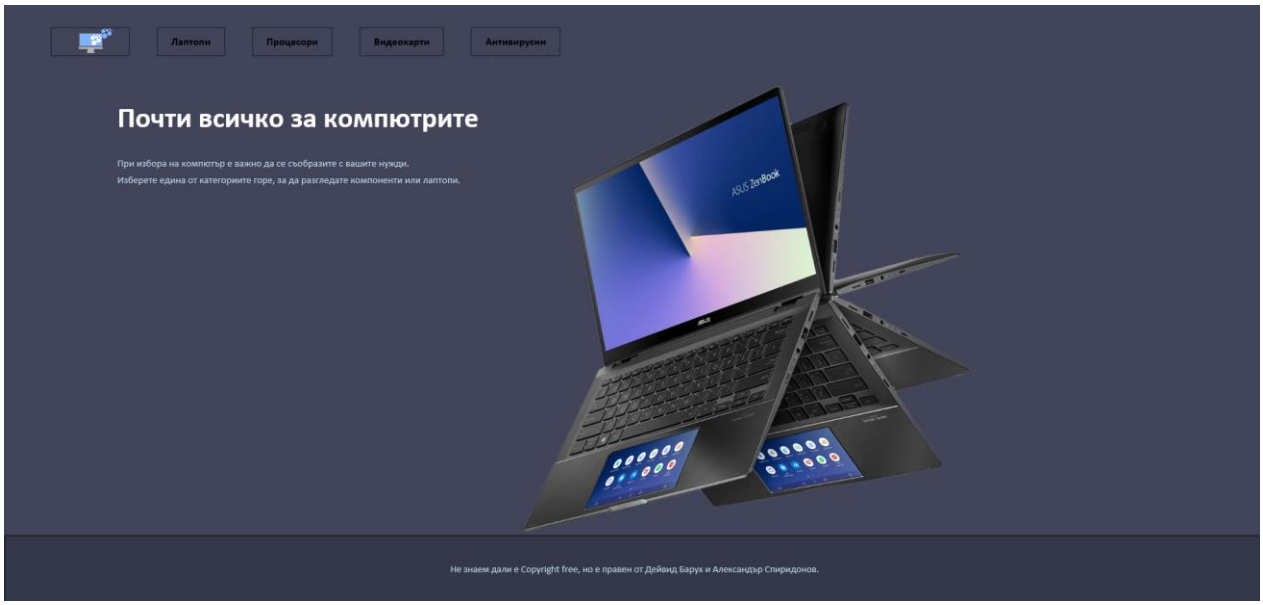
  Примерен обект видеокарта в xml файла :

```
<product iteam="antivirus" id="av3">
            <name>Intego</name>
            <malware>100</malware>
            <devices>5</devices>
            <compatibility>MAC OS,iOS</compatibility>
            <type idref="software"/>
            <purpose idref="office"/>
            <image source="antvir3_image"/>
        </product>
```

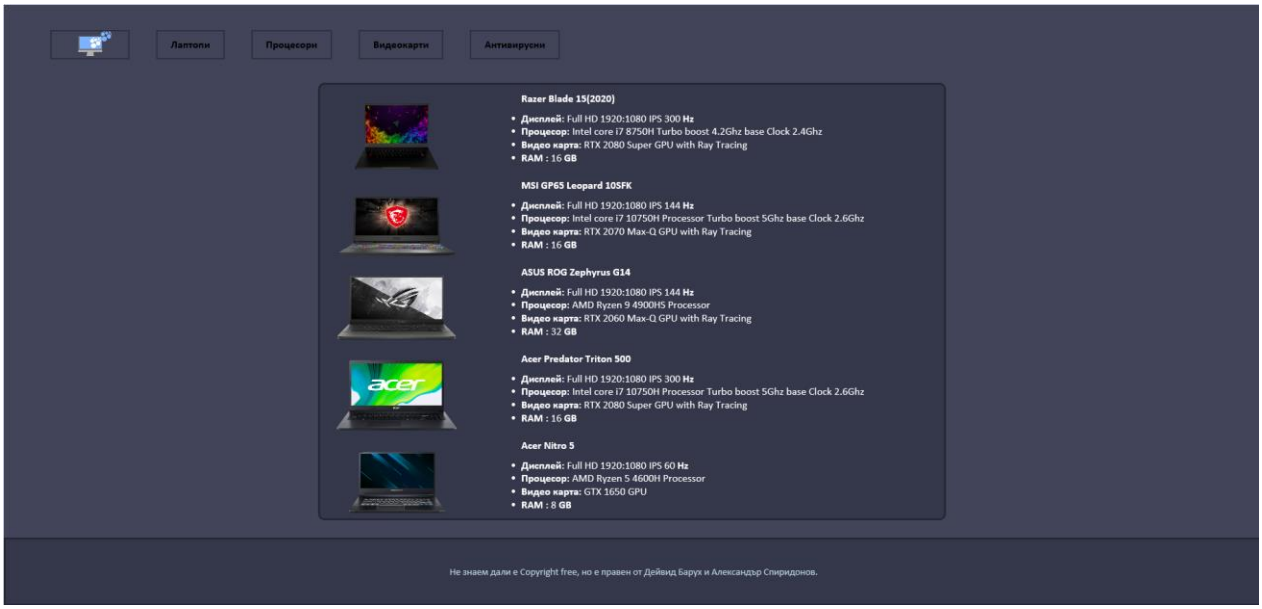
След като сме вкарали данните за продуктите в XML документа , го валидираме с XML Schema.

Завършваме с трансформация на XML документа посредством XSLT, чийто краен резултат е стилизиран HTML документ, готов за визуализация в браузър:

Главната страница :



Страница със съдържание на лаптопи:



1.2 Структура на съдържанието (2-4 стр.)

Корен на XML файла е елементът <itProducts> , който се състои от следните под елементи:

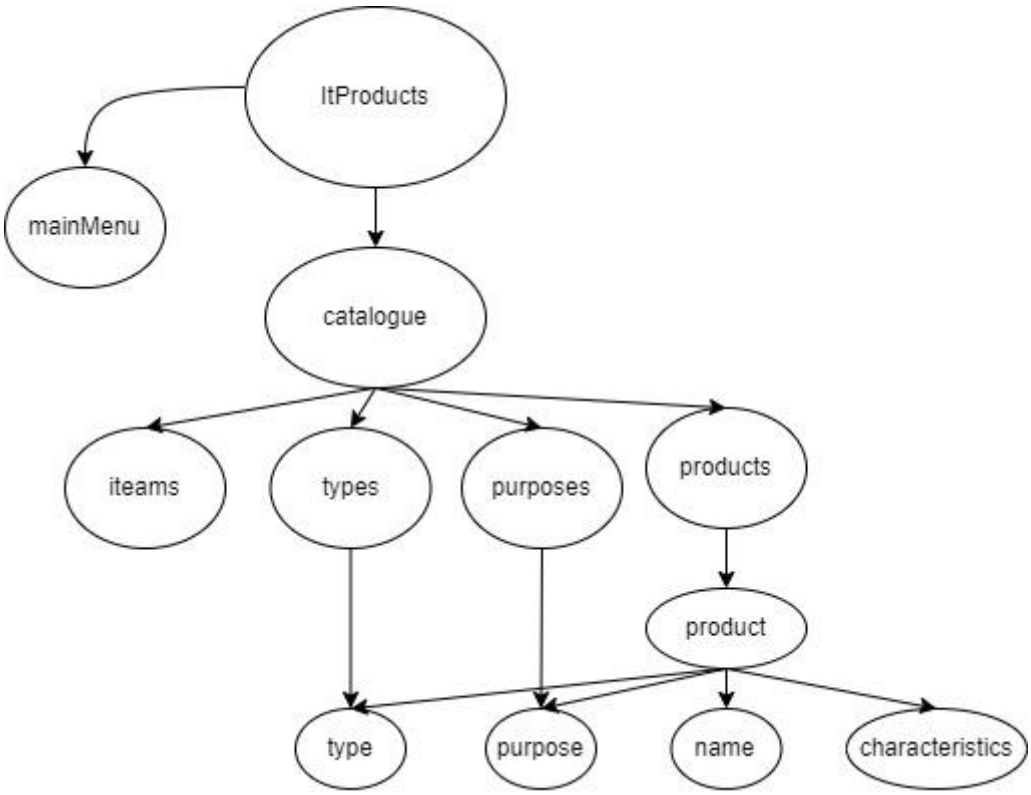
- <mainMenu> - съдържа графично съдържание, с което да се посреща потребител на визуализирания документ в браузер

- <catalog> - съдържа елементи списъци от продуктите. Елементът се съставя от следните под елементи :

    - <types> - съдържа енумерация от всички видове продукти – 2 на брой
    - <purposes> - съдържа енумерация от предназначението за всеки продукт – 2 на брой
    - <iteams> - съдържа енумерация от всички продукти – 4 на брой
    - <products> - съдържащ списък с произволна дължина от елементи <product> - 20 на брой

Всеки <product> съдържа повече от 1 под елемента, които описват характеристиките му

- *Диаграма , която представя структурата на XML съдържанието :*



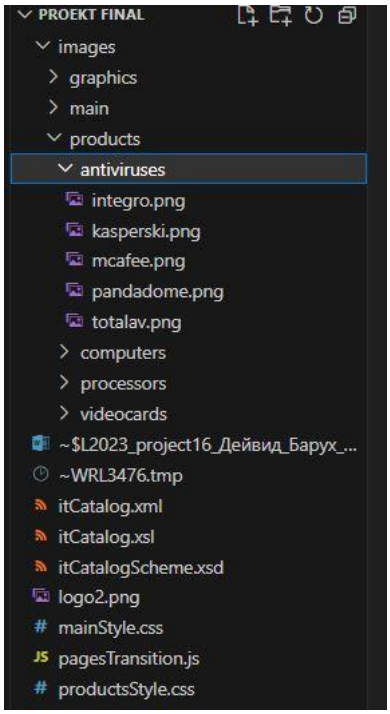
Characteristics са останалите елементи (освен името) на продукта.

- *Обобщаващ изглед към XML файла :*

```xml
<itProducts xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="itCatalogScheme.xsd">
    <mainMenu>
        <image source="main1_image"/>
    </mainMenu>

    <catalog>
        <types>
            <type id="software"/>
            <type id="hardware"/>
        </types>
        <purposes>
            <purpose id="office"/>
            <purpose id="gaming"/>
        </purposes>
        <items>
            <iteam id="processor"/>
            <iteam id="videocard"/>
            <iteam id="computer"/>
            <iteam id="antivirus"/>
        </items>
        <products>
            <product iteam="processor" id="proc1">
                <name>AMD Ryzen Threadripper 3960X </name>
                <corephis>24</corephis>
                <corelog>48</corelog>
                <frequencybase>3.8</frequencybase>
                <frequencyboost>4.5</frequencyboost>
                <cache>128</cache>
                <tdp>280</tdp>
                <type idref="hardware"/>
                <purpose idref="gaming"/>
                <image source="processor1_image"/>
            </product>
```

- *Обобщаващ изглед на съдържанието на проекта и съдържанието му :*



1.3 Тип и представяне на съдържанието (1-2 стр.)

Съдържанието на проекта се генерализирано се представя от три типа – текстово, числово и графично.

Текстовото и числовото съдържание представлява данните за продуктите, които са представени в XML файла. Конкретни параметри на продуктите, които изискват мерни единици, са представени в числов вариант като например големина на RAM или скорост на процесор. Описателните характеристики на продуктите са представени в текстов тип като например име или тип на продукта.

Графичното съдържание са снимките на продуктите и на началната страница на визуализиращият се конвертиран от XML в HTML документ . Те са в PNG формат , разположени в папката `./images` , като съответно продуктите са разположени в `./images/products` и снимката на началната страница в `./images/main`.

Изображенията са енкодирани посредством частни външни XML единици (entities) за не-XML съдържание (т.е. unparsed, бинарно). Типът им е оказан с предварително дефинирана вътрешна частна нотация PNG.

2 Дизайн (4-5 стр.)

- *XML документа*

Типовете и предназначенията са представени като отделни множества от възможни стойности, всяко от които има свой уникален идентификатор (атрибут id). В product елементи, съответния тип и регион за всеки продукт е моделиран като елементи, реферирващи съответно елементите от първоначално въведеното крайно множество от типове/ предназначения. Използван е id/idref подход.

```xml
<catalog>
    <types>
        <type id="software"/>
        <type id="hardware"/>
    </types>
    <purposes>
        <purpose id="office"/>
        <purpose id="gaming"/>
    </purposes>
    <items>
        <iteam id="processor"/>
        <iteam id="videocard"/>
        <iteam id="computer"/>
        <iteam id="antivirus"/>
    </items>
    <products>
        <product iteam="processor" id="proc1">
            <name>AMD Ryzen Threadripper 3960X </name>
            <corephis>24</corephis>
            <corelog>48</corelog>
            <frequencybase>3.8</frequencybase>
            <frequencyboost>4.5</frequencyboost>
            <cache>128</cache>
            <tdp>280</tdp>
            <type idref="hardware"/>
            <purpose idref="gaming"/>
            <image source="processor1_image"/>
        </product>
```

В документа се въвеждат и binary data (изображението за всеки продукт) като отделно entity.

```xml
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="itCatalog.xsl"?>

<!DOCTYPE itProducts [
  <!NOTATION png SYSTEM "image/png">
  <!ENTITY main1_image PUBLIC "main1_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/main/zen.png" NDATA png>

  <!ENTITY laptop1_image PUBLIC "laptop1_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/laptop_pics/razer.png" NDATA png>
  <!ENTITY laptop2_image PUBLIC "laptop2_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/laptop_pics/msi1.png" NDATA png>
  <!ENTITY laptop3_image PUBLIC "laptop3_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/laptop_pics/rog1.png" NDATA png>
  <!ENTITY laptop4_image PUBLIC "laptop4_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/laptop_pics/aspire.png" NDATA png>
  <!ENTITY laptop5_image PUBLIC "laptop5_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/laptop_pics/triton.png" NDATA png>

  <!ENTITY processor1_image PUBLIC "processor1_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/processors_pics/pn.png" NDATA png>
  <!ENTITY processor2_image PUBLIC "processor2_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/processors_pics/5950x.png" NDATA png>
  <!ENTITY processor3_image PUBLIC "processor3_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/processors_pics/10k.png" NDATA png>
  <!ENTITY processor4_image PUBLIC "processor4_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/processors_pics/i3.png" NDATA png>
  <!ENTITY processor5_image PUBLIC "processor5_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/processors_pics/thr.png" NDATA png>

  <!ENTITY vidcard1_image PUBLIC "vidcard1_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/videocards_pic/rx1.png" NDATA png>
  <!ENTITY vidcard2_image PUBLIC "vidcard2_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/videocards_pic/rx1.png" NDATA png>
  <!ENTITY vidcard3_image PUBLIC "vidcard3_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/videocards_pic/3070.png" NDATA png>
  <!ENTITY vidcard4_image PUBLIC "vidcard4_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/videocards_pic/3080.png" NDATA png>
  <!ENTITY vidcard5_image PUBLIC "vidcard5_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/videocards_pic/3090.png" NDATA png>

  <!ENTITY antvir1_image PUBLIC "antvir1_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/antiviruses/kasperski.png" NDATA png>
  <!ENTITY antvir2_image PUBLIC "antvir2_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/antiviruses/pandadome.png" NDATA png>
  <!ENTITY antvir3_image PUBLIC "antvir3_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/antiviruses/integro.png" NDATA png>
  <!ENTITY antvir4_image PUBLIC "antvir4_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/antiviruses/mcafee.png" NDATA png>
  <!ENTITY antvir5_image PUBLIC "antvir5_image_identifier" "https://raw.githubusercontent.com/davidaa30b/ITCatalog/main/images/antiviruses/totalav.png" NDATA png>
]>

<itProducts xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="itCatalogScheme.xsd">
    <mainMenu>
        <image source="main1_image"/>
    </mainMenu>

    <catalog>
```

```xml
<products>
    <product iteam="processor" id="proc1">
        <name>AMD Ryzen Threadripper 3960X </name>
        <corephis>24</corephis>
        <corelog>48</corelog>
        <frequencybase>3.8</frequencybase>
        <frequencyboost>4.5</frequencyboost>
        <cache>128</cache>
        <tdp>280</tdp>
        <type idref="hardware"/>
        <purpose idref="gaming"/>
        <image source="processor1_image"/>
    </product>
```

- *XML схемата (scheme)*

Валидацията на XML документа се осъществява чрез XML schema, намираща се във файла itCatalogSchema.xsd , реферирана с пространството от имена по подразбиране без префикс от root елемент в XML файла:

```xml
<itProducts xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="itCatalogScheme.xsd">
```

Самата валидация започва като дефинира root елемент с комплексен елемент itProducts, в който се съдържа комплексният елемент mainMenu , който отговаря за съдържанието на началната страница, и комплексният елемент catalog, който отговаря за съдържанието на продуктите. Комплексният елемент catalog от тип “catalogueType”, в чийто обхват се дефинират key/keyref елементи за валидиране на връзките продукт , тип и предназначение:

```
<xs:element name="itProducts">
  <xs:complexType>
    <xs:sequence>

      <xs:element name="mainMenu"  minOccurs="1" maxOccurs="1">
        <xs:complexType>
          <xs:sequence>
          <xs:element name="image">
            <xs:complexType>
              <xs:attribute name="source" type="xs:ENTITY" use="required"/>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:element name="catalog" type="catalogueType" minOccurs="1" maxOccurs="1">

        <xs:key name="typeKey">
          <xs:selector xpath="types/type"/>
          <xs:field xpath="@id"/>
        </xs:key>

        <xs:key name="purposeKey">
          <xs:selector xpath="purposes/purpose"/>
          <xs:field xpath="@id"/>
        </xs:key>

        <xs:keyref name="typeKeyRef" refer="typeKey">
            <xs:selector xpath="products/product/type"/>
            <xs:field xpath="@idref"/>
        </xs:keyref>

        <xs:keyref name="purposeKeyRef" refer="purposeKey">
            <xs:selector xpath="products/product/purpose"/>
            <xs:field xpath="@idref"/>
        </xs:keyref>

      </xs:element>

    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Дефиниция на самия catalogueType, в който очакваме три елемента стриктно в този ред: types, purposes, products:

```xml
<xs:complexType name="catalogueType">
    <xs:sequence>
      <xs:element ref="types"/>
      <xs:element ref="purposes"/>
      <xs:element ref="items"/>
      <xs:element ref="products"/>
    </xs:sequence>
</xs:complexType>
```

След това са дефинирани са три елемента types, purposes и iteams . Пояснението на types – очакват се неограничено множество от type елементи, всеки от които има точно един атрибут id от текстов тип и текстово съдържание. Пояснението на при purposes - очакват се неограничено множество от purpose елементи, всеки от които има точно един атрибут id от текстов тип и текстово съдържание. Пояснението на при iteams - очакват се неограничено множество от iteam елементи, всеки от които има точно един атрибут id от текстов тип и текстово съдържание. Всички тези елементи са от комплексен тип с името ` idTypeOnly `.

```xml
<xs:element name="types">
  <xs:complexType>
    <xs:sequence>
      <xs:element
        name="type" minOccurs="2" maxOccurs="unbounded"
        type="idTypeOnly"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="purposes">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="purpose" minOccurs="2" maxOccurs="unbounded"
      type="idTypeOnly"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="items">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="iteam" minOccurs="2" maxOccurs="unbounded"
      type="idTypeOnly"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:complexType name="idTypeOnly">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="id" type="xs:ID" use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

Накрая на схемата се намира и дефиницията на елемента products като комплексен тип, който очаква поредица от поне един ресторант. Типът на елемента product е изнесен в отделен

комплексен тип с името productType, в който вече дефинираме конкретната последователност от елементи, в който се включват :

-name от тип произволен текст (xs:string)

-display от тип произволен текст (xs:string)

-displayhertzs от тип произволен текст (xs:integer)

-processor от тип произволен текст (xs:string)

-gpu от тип произволен текст (xs:string)

-ram от тип произволен текст (xs:integer)

-memory от тип произволен текст (xs:integer)

-memoryspeed от тип произволен текст (xs:string)

-bandwidth от тип произволен текст (xs:integer)

-corephis от тип произволен текст (xs:integer)

-corelog от тип произволен текст (xs:integer)

-frequencybase от тип произволен текст (xs:float)

-frequencyboost от тип произволен текст (xs:float)

-cache от тип произволен текст (xs:string)

-tdp от тип произволен текст (xs:string)

-malware от тип произволен текст (xs:integer)

-devices от тип произволен текст (xs:integer)

-type - от тип произволен текст (onlyidref)

-purpose - от тип произволен текст (onlyidref)

-image от тип, съдържащ единствен атрибут source

Само елементите name, type, purpose и image са задължителни, като останалите елементи се избират в зависимост от искания продукт за представяне (лаптоп, видеокарта, процесор или антивирусна програма)

Типът onlyidref е отделен тип, изискващ празен елемент с единствен атрибут idref.

```
<xs:complexType name="productType">
  <xs:sequence>
    <xs:element name="name" type="xs:string" minOccurs="1"/>
    <!--computerType-->

    <xs:element name="display" type="xs:string" minOccurs="0"/>
    <xs:element name="displayhertzs" type="xs:integer" minOccurs="0"/>
    <xs:element name="processor" type="xs:string" minOccurs="0"/>
    <xs:element name="gpu" type="xs:string" minOccurs="0"/>
    <xs:element name="ram" type="xs:integer" minOccurs="0"/>

    <!--videocardsType-->

    <xs:element name="memory" type="xs:integer" minOccurs="0"/>
    <xs:element name="memoryspeed" type="xs:string" minOccurs="0"/>
    <xs:element name="bandwidth" type="xs:integer" minOccurs="0"/>

    <!--processorsType-->

    <xs:element name="corephis" type="xs:integer" minOccurs="0"/>
    <xs:element name="corelog" type="xs:integer" minOccurs="0"/>
    <xs:element name="frequencybase" type="xs:float" minOccurs="0"/>
    <xs:element name="frequencyboost" type="xs:float" minOccurs="0"/>
    <xs:element name="cache" type="xs:string" minOccurs="0"/>
    <xs:element name="tdp" type="xs:string" minOccurs="0"/>

    <!--antivirusType-->
    <xs:element name="malware" type="xs:integer" minOccurs="0"/>
    <xs:element name="devices" type="xs:integer" minOccurs="0"/>
    <xs:element name="compatibility" type="xs:string" minOccurs="0"/>

    <xs:element name="type" type="onlyidref"/>
    <xs:element name="purpose" type="onlyidref"/>
    <xs:element name="image">
      <xs:complexType>
        <xs:attribute name="source" type="xs:ENTITY" use="required"/>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:ID" use="required"/>
  <xs:attribute name="iteam" type="xs:string" use="required"/>
</xs:complexType>

<xs:complexType name="onlyidref">
  <xs:attribute name="idref" type="xs:IDREF" use="required"/>
</xs:complexType>
```

- *XSL документа (трансформиращ XML документа до HTML страница)*

Трансформацията на XML документа до HTML страница се извършва чрез XSL stylesheet финален. В него съдържанието е представено с HTML таблици, в които се показват продуктите. Изплозван е

CSS за преобразуване на страницата в по-приятен вид и един JS скрипт, спомагащ за динамично преминаване през страници, в които са представени отделните продукти.

В началото на XSL файла са дефинирани параметрите : sortOnComputer, sortOrderComputer, sortTypeComputer, sortOnVideocards, sortOrderVideocards, sortTypeVideocards, sortOnProcessors, sortOrderProcessors, sortTypeProcessors, sortOnAntiviruses, sortTypeAntiviruses, sortOnAntiviruses. Те се изплозват за сортировката от таблици за всеки продукт.

sortOn… - задава по какво ще сортираме таблица от даден продукт (име на елемент от продукт или някой негов параметър).

sortDataType… - задава типа на данните, по които ще сортираме (в нашето решение, това е text или number)

sortOrder… - задава реда на подредбата (ascending / descending)

Създадени са 9 такива параметъра, за да може да се поддържа отделна сортировка за всеки продукт.

След тях е дефиниран XSL ключ , който служи за индексация на всички елементи с наличен ключ по техния id атрибут.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <xsl:output method="html"/>


    <xsl:param name="sortOnComputer">name</xsl:param>
    <xsl:param name="sortOrderComputer">descending</xsl:param> <!--descending  ascending-->
    <xsl:param name="sortTypeComputer">text</xsl:param>

    <xsl:param name="sortOnVideocards">name</xsl:param>
    <xsl:param name="sortOrderVideocards">descending</xsl:param>
    <xsl:param name="sortTypeVideocards">text</xsl:param>


    <xsl:param name="sortOnProcessors"></xsl:param>
    <xsl:param name="sortOrderProcessors">descending</xsl:param>
    <xsl:param name="sortTypeProcessors">text</xsl:param>


    <xsl:param name="sortOnAntiviruses">name</xsl:param>
    <xsl:param name="sortOrderAntiviruses">descending</xsl:param>
    <xsl:param name="sortTypeAntiviruses">text</xsl:param>


    <xsl:key name="purposeId" match="catalog/purposes/purpose" use="@id"/>
```

След дефинираните параметри и ключ се представя шаблона генериращ HTML документа. Започваме от корена на XML дървото. В head – а на HTML документа прилагаме CSS, който ще служи за оформяне на изгледа на проекта. Той се състои от два файла : mainStyle.css(служи за

оформлението на заглавната страница) и productsStyle.css (служи за оформлението на страниците от продуктите). После се прилага JAVASCRIPT файл pagesTransition.js, който служи за преминаването през страниците в проекта. В тялото на генерираната HTML страница се прилага меню от бутони, което имплементира JAVASCRIPT кода от pagesTransition.js , като прилага логика към всеки бутон, чрез която ще се изпълнява преминаването към друга страница при натискане на бутона. Бутоните са оформени в поле с CSS, който ги поставя на върха на всяка страница.

```
<xsl:template match="/">
    <html>
        <head>
            <title>Почти всичко за компютрите</title>
            <link rel = "stylesheet" type="text/css" href="mainStyle.css"/>
            <link rel = "stylesheet" type="text/css" href="productsStyle.css"/>

        </head>
        <script type="text/javascript" src="pagesTransition.js"></script>
        <body>
            <div class = "conta">
                <div class =  "menu">
                    <ul>
                        <li class = "logo">
                            <button class="transparent-button" onclick="showMainmenu();">
                                <img src = "logo2.png"/>
                            </button>
                        </li>
                        <li>
                            <button class="transparent-button" onclick="showComputers();">
                                Лаптопи
                            </button>
                        </li>

                        <li>
                            <button class="transparent-button" onclick="showProcessors();">
                                Процесори
                            </button>
                        </li>

                        <li>
                            <button class="transparent-button" onclick="showVideocards();">
                                Видеокарти
                            </button>
                        </li>

                        <li>
                            <button class="transparent-button" onclick="showAntiviruses();">
                                Антивирусни
                            </button>
                        </li>
                    </ul>
                </div>
                <xsl:apply-templates/>
            </div>


            <div class = "bottompiece">
                <div class = "dolu">Не знаем дали е Copyright free, но е правен от Дейвид Барух и Александър Спиридонов.</div>
            </div>
        </body>
    </html>
</xsl:template>
```

След полето от бутони се прилагат шаблони за съдържанието на страницата, което се постига чрез <xsl:apply-templates>.Шаблоните са за начална страница и за продуктите.

Шаблонът за началната страница:

```
<xsl:template match="/itProducts/mainMenu">
    <div id="mainmenuContainer" style="display:inline">
        <div class = "banner">
            <div class="app">
                <h1>Почти всичко за компютрите</h1>
                <p>
                    При избора на компютър е важно да се съобразите с вашите нужди.
                    <br/>
                    Изберете една от категориите горе, за да разгледате компоненти или лаптопи.
                </p>

            </div>
            <img class="main-image" src="{unparsed-entity-uri(image/@source)}" />
        </div>
    </div>
</xsl:template>
```

В него има таг със id , което се използва в JAVASCRIPT файла за оказване коя страница да се визуализира. Използва се XPath функцията unparsed-entity-uri, за да извлечем идентификатора от самото entity, която се реферира от @source атрибутът на елемента, за да визуализираме снимката на началната страница.

В шаблона за продуктите има 4 таг с id(съответно за лаптопи, процесори, видеокарти и антивирусни системи) ,които се използват от JAVASCRIPT файла. Във всеки такъв таг се използва таблица <table> за представяне на снимково и текстово съдържание на всеки вид продукт . Обхожда се всеки продукт чрез xsl:for-each като накрая на подадения път, за да се филтрира за всеки вид продукт се отбелязва с на елемента product атрибута iteam да е равно на този вид(computer, videocard, processor, antivirus) .

```
<xsl:template match="/itProducts/catalog/products">

<div id="computerContainer" style="display:none;">
    <table cellspacing="0" width="50%" bgcolor="#35384B" class="tt">
        <xsl:for-each select="/itProducts/catalog/products/product[@iteam='computer'] ">
            <xsl:sort select="*[name(.) = $sortOnComputer]" data-type="{$sortTypeComputer}" order="{$sortOrderComputer}"/>
            <tr>
                <td>
                    <img class="computer-image" src="{unparsed-entity-uri(image/@source)}" />
                </td>
                <td>
                    <ul class="ma">
                        <div class="sp">
                            <b>
                                <xsl:value-of select="name"/>
                            </b>
                        </div>

                        <li>
                            <b>Дисплей: </b><xsl:value-of select="display"/>
                            <xsl:value-of select="displayhertzs"/> <b> Hz </b>
                        </li>
                        <li>
                            <b>Процесор: </b><xsl:value-of select="processor"/>

                        </li>
                        <li>
                            <b>Видео карта: </b><xsl:value-of select="gpu"/>
                        </li>
                        <li>
                            <b>RAM : </b><xsl:value-of select="ram"/><b> GB</b>
                        </li>
                        <li>
                            <b>Предназначение: </b> <xsl:value-of select="key('purposeId', purpose/@idref)/@id"/>
                        </li>
                    </ul>
                </td>
            </tr>
        </xsl:for-each>
    </table>
</div>
```

След това следва визуализацията на продуктите с неговите характеристики и графично съдържание.Като това се прилага и за останалите продукти.

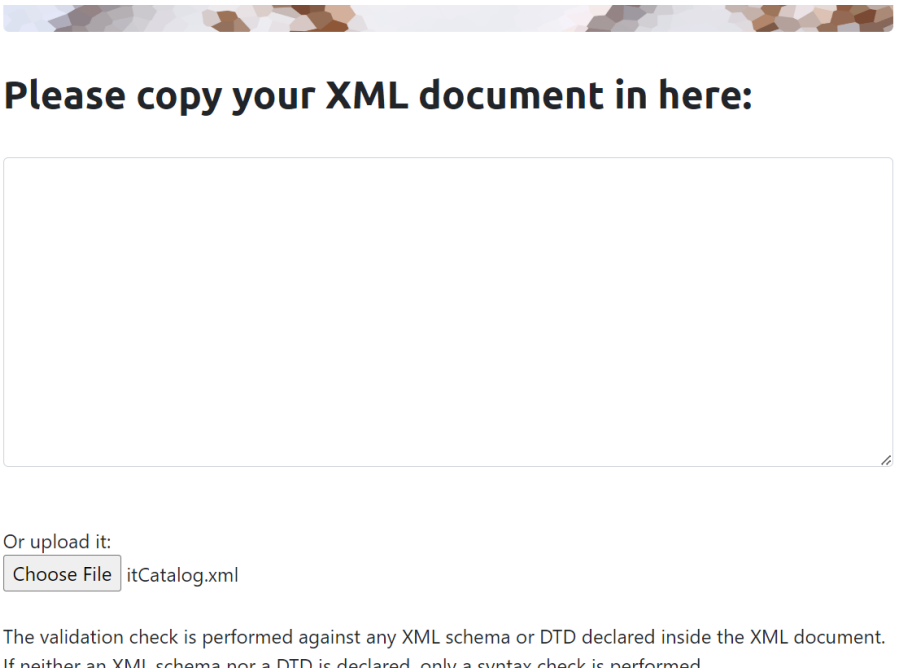
3 Тестване (2-3 стр.)

За тестване на XML файла и XML схемата са използвани онлайн сайтовете : Liquid Technologies и https://www.xmlvalidation.com/

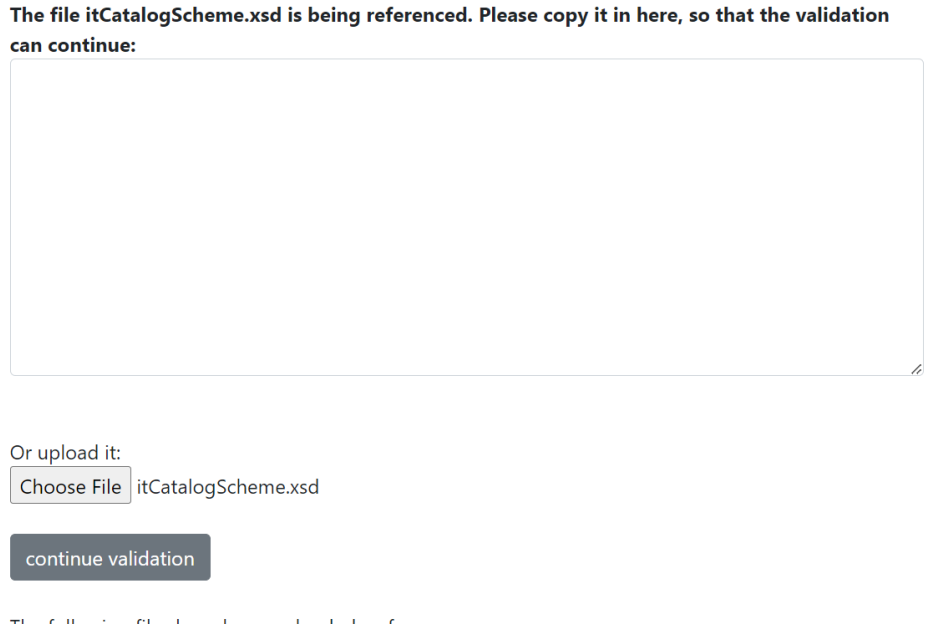
За тестване на функционалността на сайта използвахме Internet Explorer и Microsoft Edge.

Резултатите от тестването на XML файла и XML схемата чрез https://www.xmlvalidation.com/ са следните :

Вкарване на XML файла:



Вкарване на XML схемата:



Резултатът от валидацията:

No errors were found

The following files have been uploaded so far:
XML document:
itCatalogScheme.xsd
Click on any file name if you want to edit the file.

4 Заключение и възможно бъдещо развитие (1-2 стр.)

В заключение, може да кажем, че се получи един добре изглеждащ HTML сайт, като сме използвали добри практики за работа с XML файлове , XML схема валидираща XML файла и XSL файл, който преобразува XML файла до HTML страница.

В бъдеще екипът ни има амбицията да добави още повече продукти. Да се използват снимките като линкове към сайтове, където съответните продукти могат да се намерят на най-добра цена. Създаване и на 3-D модели, които могат да се видят чрез натискане на продукта. Да се добави възможност, чрез която да могат да се добавят допълните продукти към XML файла директно през сайта. Искаме да приложим динамично сортиране на таблиците от различните продукти. Да го оптимизираме да работи на повече браузери.

5 Разпределение на работата

- съставяне на структурата на входния XML файл - Дейвид;
- изготвяне на XSD валидационна схема - Дейвид;
- изготвяне на XSLT трансформация – съвместна работа;
- допълнителен JS скрипт за динамично сортиране - Дейвид;
- CSS stylesheet - Александър;
- Оптимизиране и тестване на крайната версия на проекта - Александър;
- изготвяне на документация - съвместна работа.

6 Използвани литературни източници и Уеб сайтове

- https://stackoverflow.com/ - за логика на XML схема и XSLT templates
- https://www.liquid-technologies.com/online-xml-validator - за тестване на XML файла с данни и XML схемата

- https://www.w3schools.com/xml/schema_intro.asp за логика на XML схема и XSLT templates
- https://www.ozone.bg/ - избор на продукти
- https://ardes.bg/?gad_source=1&gclid=CjwKCAiA-vOsBhAAEiwAIWR0TfOBw0fOwmFhmBrouNzUfHtUMf0rEppqdpYEq9FSQRv6YLZtuYbB3BoCVdMQAvD_BwE избор на продукти
- материалите от курса в moodle