



**Софийски университет „Св. Кл.
Охридски“**

Факултет по математика и информатика



*Бакалавърска програма
„Софтуерно инженерство“*

Предмет: XML технологии за семантичен Уеб

Зимен семестър, 2022/2023 год.

Тема 25: Каталог на язовирите в България

Курсов проект

Автори:

Харут Партамиан, фак. номер 62560

Емилиан Спасов, фак. номер 62617

януари, 2023 г.

София

Съдържание

1 Въведение	3
2 Анализ на решението	4
2.1 Работен процес	4
2.2 Структура на съдържанието	5
2.3 Тип и представяне на съдържанието	7
3 Дизайн	8
4 Тестване	10
5 Заключение и възможно бъдещо развитие	11
6 Разпределение на работата	12
7 Използвани литературни източници и Уеб сайтове	13

1 Въведение

Изборът ни на тема се мотивира от значението и сложността спрямо останалите теми, тъй като не е от най-лесните, нито най-трудните. Темата ни се стори от изключителна важност, тъй като разнообразието от язовири в България е голямо, но в същото време слабо известно на широката аудитория.

Както споменахме, чрез нашия проект се стремим да обогатим общата култура на потребителите и читателите относно водохранилищата в България, като покажем техни различни качества и характеристики, по интересен и лесен за разбиране начин. Забелязали сме тенденцията, че голяма част от гражданите са слабо запознати с водните природни богатства на България. Затова сметнахме за подходящо да разработим тази тема с включена визуализация, чрез която още по-лесно и интересно ще се опознаят част от язовирите на територията на нашата Република.

За реализацията на решението на посочения проблем сме използвали различни технологии и средства изучавани в курса “XML технологии за семантичен уеб”, като xml, xslt, css и други файлове. На страница 2 е показано съдържание, което описва структурата на документа, като основните точки, с които ще се запознае читателят на този документ, са анализ на имплементираното решение, дизайн на кода и изгледа на страницата, тестване през различни браузъри, заключение, резюмиращо проекта и точка, в която се описват използваните източници.

2 Анализ на решението

2.1 Работен процес

Работният процес, като част от анализа на решението, се изразява в извличането на данни под формата на снимки в .jpg формат и .xml файл, в който е описана основната структура на обектите, използвани в проекта.

Обработката на “суровите” данни се случва чрез .xslt файл и xsl темплейти.

Третата стъпка от работния процес - изходът се състои в съставянето на html и css, който се рендерира в браузъра. Това съдържание трябва да бъде представено и използвано от потребителите на проекта за образователни цели.

На следващите снимки ще покажем част от .xml (структурата за даден язовир), част от .xslt (обработката на данните) и визията на проекта в браузър.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="damsXSLT.xsl"?>
<damsCatalog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="damsCatalogSchema.xsd">
  <damsList>
    <dam>
      <id>1</id>
      <name>Язовир Батак</name>
      <region>област Пазарджик</region>
      <regionId>1</regionId>
      <altitude>1107,8 м</altitude>
      <constructionYear>1959 г.</constructionYear>
      <damWall>
        <type>Земнонасипна с глинен зъб</type>
        <height>35 м</height>
        <length>273 м</length>
      </damWall>
      <lake>
        <area>21.4</area>
        <full-area>21,4 км2</full-area>
        <waterVolume>310 000 000 м3</waterVolume>
        <waterSupplyPoolArea>463,29 км2</waterSupplyPoolArea>
      </lake>
      <picture location="pic1"/>
    </dam>
  </damsList>
</damsCatalog>
```


Язовири в България

Язовири

По площ

По област

Язовир Кърджали



Име

Язовир Кърджали

Обем

497 200 000 m³

Област

област Кърджали

Площ

13,4 km²

Надморска височина

331,45 m

Вид стена

Бетонно дъгово-гравитачна

Година на откриване

1977г.

```

<xsl:template match="/damsCatalog/damsList">
  <div id="damsListContainer" class="container">
    <div class="row">
      <xsl:for-each select="/damsCatalog/damsList/dam[id < 10]">
        <xsl:sort
          select="id"
          data-type="number"
          order="descending" />
        <div style="color:white;" class="dam col-lg-12 container"...>
      </xsl:for-each>
    </div>
  </div>

  <div id="areaContainer" style="display:none;" class="container"...>
</xsl:template>

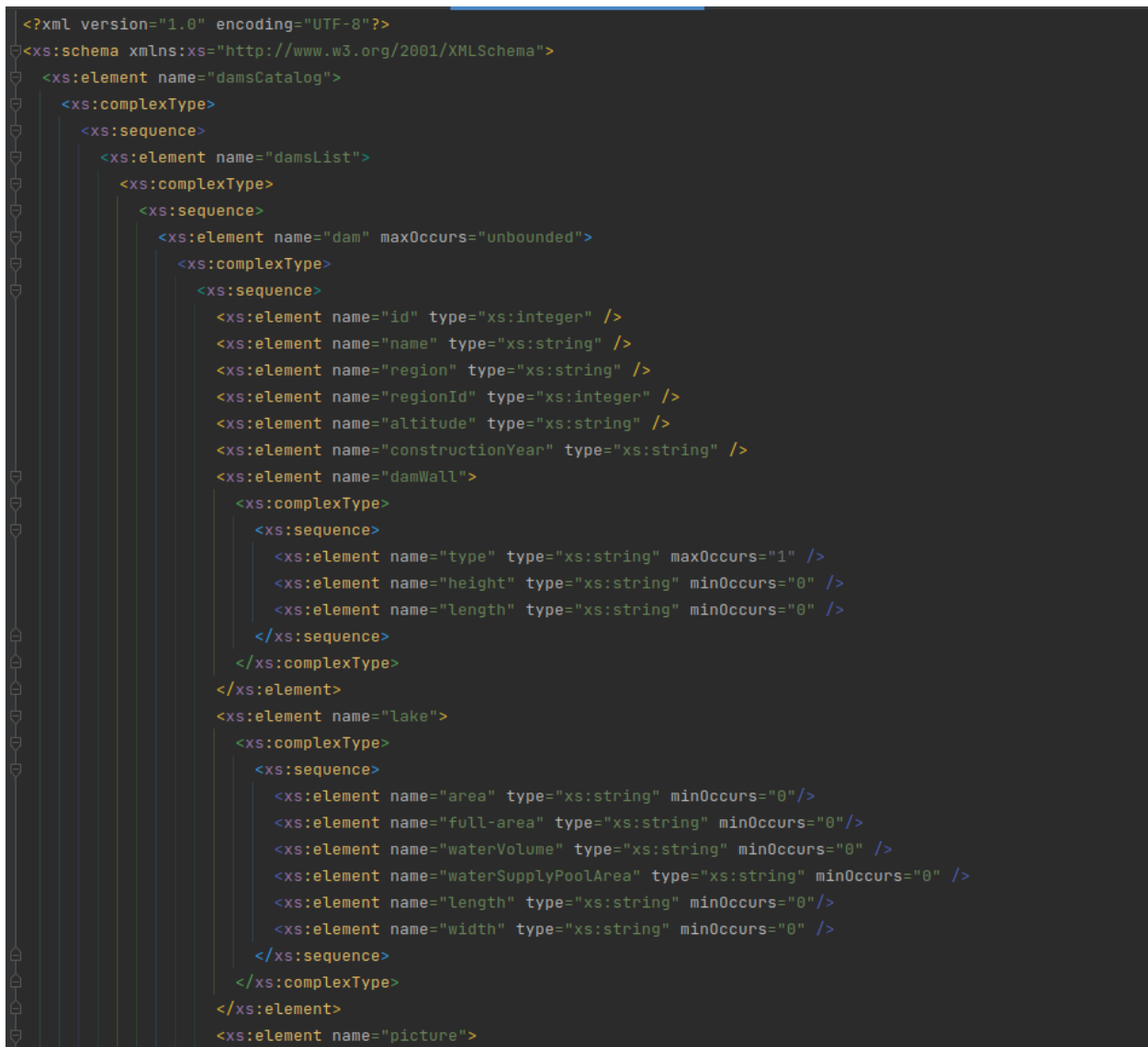
```

2.2 Структура на съдържанието

XML документа започва с таг, който показва версията на xml, която използваме. След това основната част е тагът “damsList”, в който изброяваме всеки язовир в собствен таг “dam”. Всеки язовир се описва чрез следните вложени характеристики: “id” (цяло число), “name” (символен низ), “region” (символен низ), “regionId” (цяло число), “altitude” (символен низ), “constructionYear” (символен низ), “damWall”, “lake” с вложени характеристики “area” (число с плаваща запетая), “full-area” (символен низ), “waterVolume” (символен низ), “waterSupplyPoolArea” (символен низ); picture и други. В края на файла е поставен списък от

региони, вложени под общ таг “regions”, които реферират към всеки регион и служат за имплементацията на сортиращата и групиращата функционалност.

Прилагаме снимки под формата на screenshot:



```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="damsCatalog">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="damsList">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="dam" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="id" type="xs:integer" />
                    <xs:element name="name" type="xs:string" />
                    <xs:element name="region" type="xs:string" />
                    <xs:element name="regionId" type="xs:integer" />
                    <xs:element name="altitude" type="xs:string" />
                    <xs:element name="constructionYear" type="xs:string" />
                    <xs:element name="damWall">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="type" type="xs:string" maxOccurs="1" />
                          <xs:element name="height" type="xs:string" minOccurs="0" />
                          <xs:element name="length" type="xs:string" minOccurs="0" />
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                    <xs:element name="lake">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="area" type="xs:string" minOccurs="0"/>
                          <xs:element name="full-area" type="xs:string" minOccurs="0"/>
                          <xs:element name="waterVolume" type="xs:string" minOccurs="0" />
                          <xs:element name="waterSupplyPoolArea" type="xs:string" minOccurs="0" />
                          <xs:element name="length" type="xs:string" minOccurs="0"/>
                          <xs:element name="width" type="xs:string" minOccurs="0" />
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                    <xs:element name="picture">
```

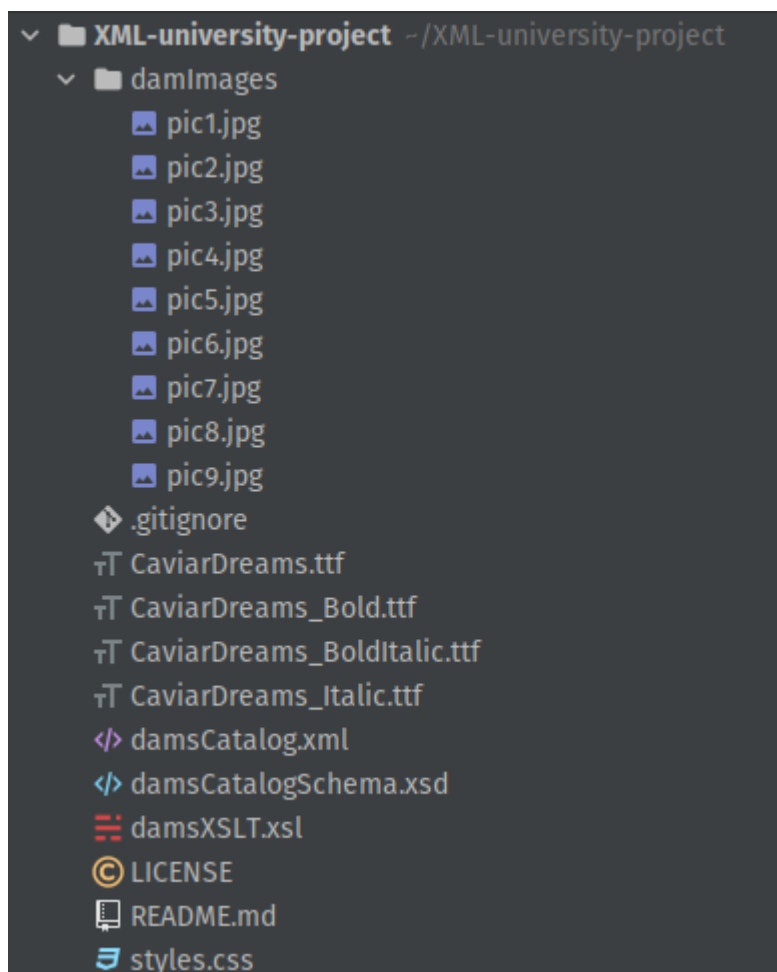
```

    </xs:complexType>
  </xs:element>
  <xs:element name="picture">
    <xs:complexType>
      <xs:attribute name="location" type="xs:ENTITY"/>
    </xs:complexType>
  </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="regions" maxOccurs="1">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="regionInfo" maxOccurs="unbounded">
        <xs:complexType>
          <xs:simpleContent>
            <xs:extension base="xs:string">
              <xs:attribute type="xs:string" name="regionRef" />
              <xs:attribute type="xs:integer" name="regionRefId" />
            </xs:extension>
            </xs:simpleContent>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>

```

2.3 Тип и представяне на съдържанието

За целите и реализацията на проекта сме използвали текстово и графично съдържание, разделено във файлове с различен формат. По този начин образуваме файловата структура на проекта и той става удобен за четене, компилиране и редактиране. Имаме 9 графични изображения във .jpg формат събрани в директорията damImages, 4 текстови документа - damsCatalog.xml с размер 6.53 килобайта, damsCatalogSchema.xsd с размер 3.11 килобайта, damsXSLT.xsl с размер 9 килобайта, styles.css с размер 2.53 килобайта и 4 css визуални изгледи с формат .ttf, с общ размер 240 килобайта. На снимката, която ще приложим се забелязват и други файлове, но те са строго специализирани за употребата на проекта в система за следене на версиите (Github).



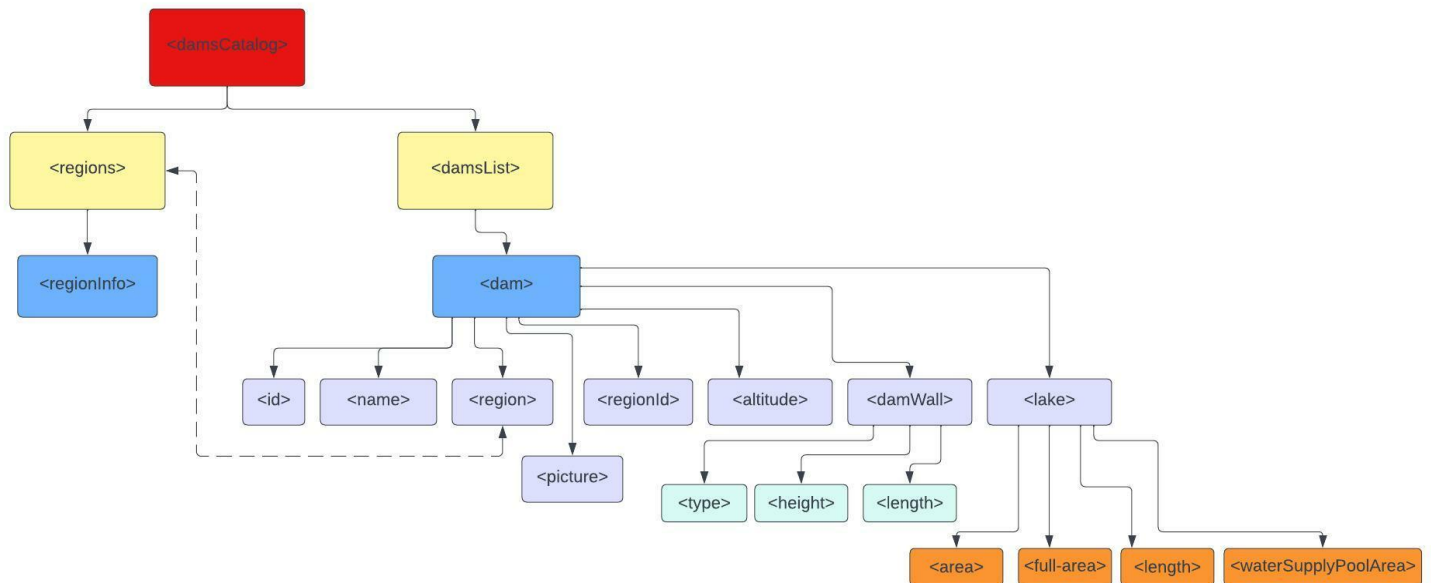
3 Дизайн

Както сме описали в точка 2.2 в XML документа използваме няколко нива на влагане за реализация на описанието на язовирите и регионите, като връзката на регионите с язовирите става посредством идентификаторите `regionId` и `regionRefId`. За валидация влагаме `xsd` файла, чиято структура и визия се вижда на снимките в точка 2.2, чрез следния таг:

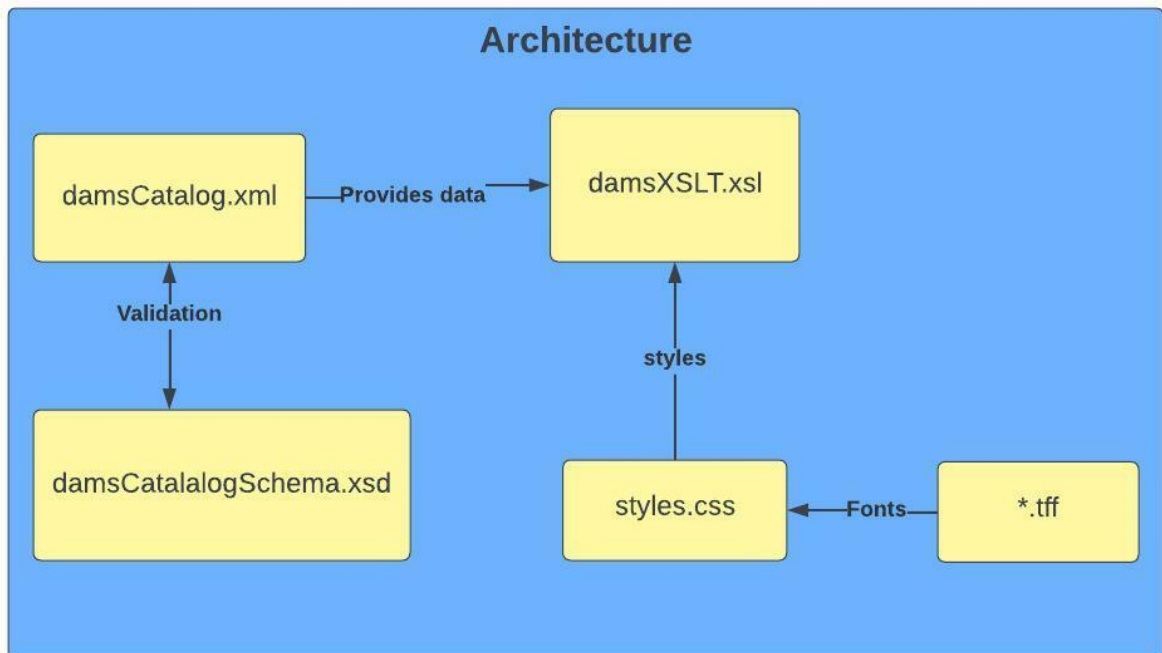
```
<damsCatalog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="damsCatalogSchema.xsd">
```

За трансформацията на данни и визуализацията им в браузъра използваме `.xsl`. Използваме контролни тагове като `<xsl-template>` и `<xsl-for-each>` за обхождане на списъка от язовири, предоствен от `damsCatalog.xml`. При обхождане на водохранилищата за всяко от свойствата им създаваме подходящ `html` таг, чрез който да бъдат рендерирани в браузъра. За подобряване на потребителското преживяване сме използвали подходящи `css` свойства за стилизиране на страницата. Използвали сме два различни вида сортировки (по

релевантност и по площ) - чрез `<xsl:sort>`, и една групировка за язовирите (по област) - чрез използване на `regions` структурата от xml документа.



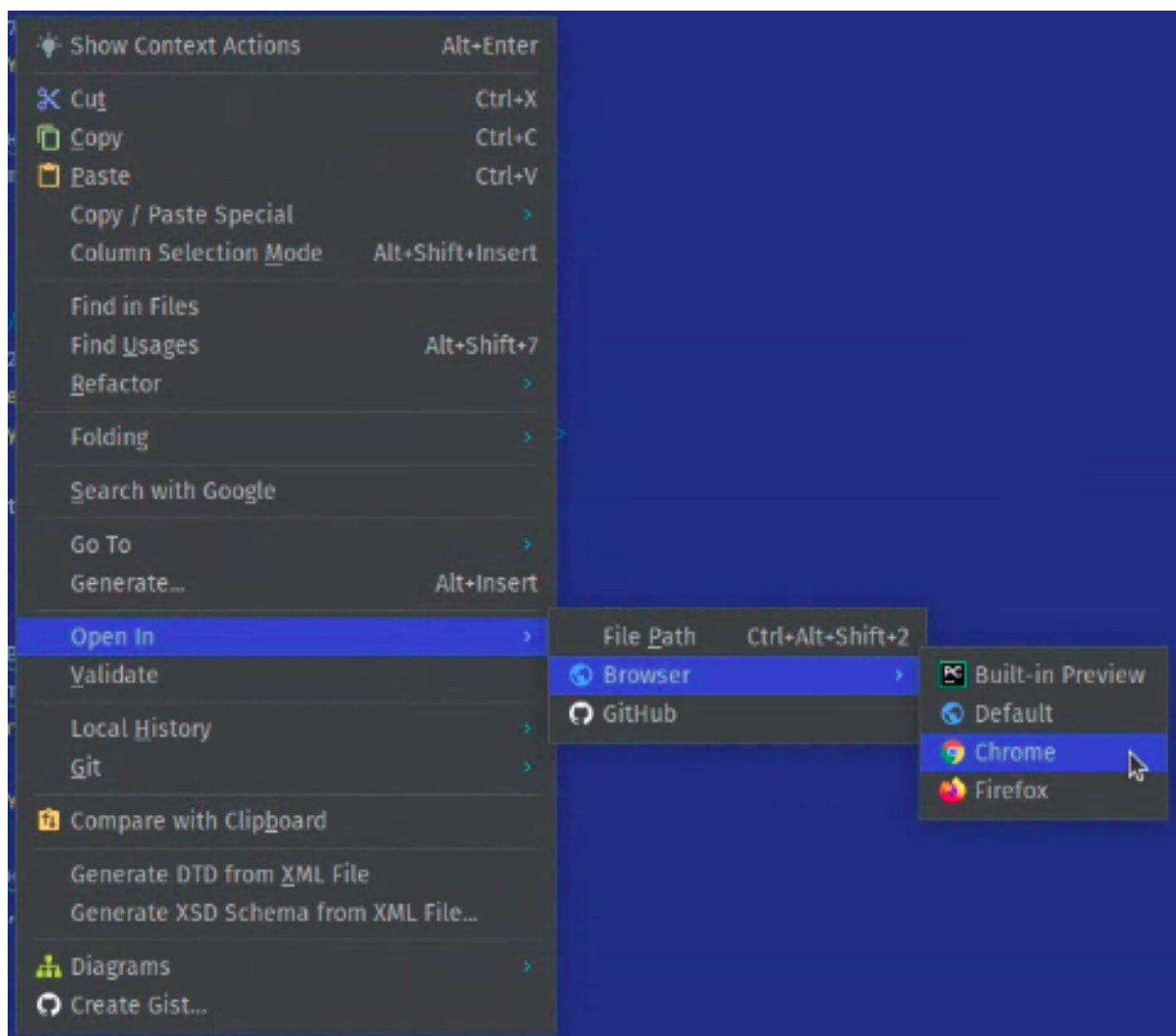
Архитектура на проекта:





4 Тестване

За разработка и тестване на проекта използвахме програмна среда (IDE и текстов редактор), чрез която да заредим проекта в различни браузъри (Chrome, Firefox). Тествахме го и през терминал чрез зареждане на HTTP сървър с помощта на python. Прилагаме екранни снимки:

```
~/XML-university-project main .....  
> python -m http-server 8080
```



Резултатът от всяко изпълнение на зареждане е следния:

Язовири в България		
Язовири	По площ	По област
Язовир Кърджали		
	Име	Обем
	Язовир Кърджали	497 200 000 m³
	Област	Площ
	област Кърджали	13,4 km²
Надморска височина		Вид стена
331,45 m		Бетонно дъгово-гравитачна
Година на откриване		
1977г.		
Язовир Ивайловград		
	Име	Обем
	Язовир Ивайловград	156 700 000 m³
	Област	Площ
	област Хасково	15,1 km²

5 Заключение и възможно бъдещо развитие

Като заключение бихме казали, че проекта беше интересен за разработване, както по отношение на функционалност, така и на визия. Като предимства на използването на XML бихме казали, че той е гъвкав и разширяем, което означава, че може да се използва за кодиране на голямо разнообразие от типове данни и структури. Също така голям бонус е че е независим от платформата, което означава, че може да се използва на всяка операционна система или устройство и има силна поддръжка от широк набор от софтуерни инструменти, включително парсери, редактори и библиотеки, което улеснява работата с него. XML има и минуси, като част от тях са:

- XML може да бъде многословен, което означава, че може да изисква много данни за представяне на прости концепции. Това може да го направи по-малко ефективен за предаване и съхранение в сравнение с други формати.
- XML документите могат да бъдат трудни за четене и разбиране от хората, особено ако са големи или сложни.
- XML има строги синтактични правила, които трябва да се спазват, за да може документът да се счита за добре оформен и структуриран. Това може да затрудни научаването и използването на начинаещите.
- Анализът на XML може да бъде бавен, особено за големи документи. Това може да е проблем за приложения, които трябва да обработват много данни в реално време.

Като алтернативи на използването на XML бихме предложили:

- JSON (JavaScript Object Notation) – JSON е лек формат за обмен на данни, който се основава на подмножество на JavaScript. Той е лесен за четене и писане и често се използва като алтернатива на XML в уеб базирани приложения.
- CSV (стойности, разделени със запетая) - CSV е прост, обикновен текстов формат, който се използва за съхраняване на таблични данни (като електронна таблица). С него се работи лесно и може да се отваря и редактира във всеки текстов редактор или програма за електронни таблици.
- YAML (YAML не е език за маркиране) - YAML е четим формат за сериализиране на данни, който често се използва за конфигурационни файлове и съхранение на данни. Той е лесен за четене и писане и е особено подходящ за съхранение на йерархични данни.
- Протоколни буфери - Протоколните буфери (protobufs) са двоичен сериализиращ формат, разработен от Google. Те са проектирани да бъдат малки, ефективни и лесни за работа и често се използват за обмен на данни в разпределени системи.
- BSON (двоичен JSON) - BSON е двоичен сериализиращ формат, който е базиран на JSON. Използва се за съхраняване на данни по по-ефективен начин от JSON и често се използва в системи за бази данни.
- Pickle - Pickle е специфичен за Python сериализиращ формат, който се използва за съхраняване на обекти на Python в поток от байтове. Не е предназначен да се използва като формат за сериализация с общо предназначение, но често се използва за съхраняване на данни в приложения на Python.

Ще завършим като кажем, че като бъдещо развитие към проекта можем да добавим още язовири и допълнителни характеристики към тях. От друга страна може да не е само за язовири, а да добавим езера и реки и всякакви водни източници. Може да направим разделение по градове и да направим отделни секции за всеки град и област и още много други неща.

6 Разпределение на работата

Работата беше разпределена равномерно, за да няма ощетяване на някой от участниците в екипа, по следния начин - разработване на XML документа и валидацията чрез XSD - Харут Партамиан, а XSL и CSS документите, както и подпомагащите ги файлове - Емилиан Спасов. След като всеки завърши частта, която му бе заложена за изпълнение, заедно сглобихме и влагахме файловете, така че да може проекта да има финалната си визия.

7 Използвани литературни източници и Уеб сайтове

1. <https://www.w3schools.com/>
2. <https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/xml/doc/>
3. Презентациите на лекциите към курса
4. Файловете използвани по време на упражненията към курса