

Курсовият проект е на тема „Ресторант” – описанието ще включва име на ресторанта, кратко описание в рамките на 1-2 изречения, с които ресторантът се рекламира, адрес, телефон, служителите на ресторанта и тайни от кухнята му.

Проектът се състои от:

- `restaurant.dtd` - файл, който е описание на елементите и атрибутите в xml файловете .
- `restaurantX.xml` – 5 валидни за това DTD xml-файла (X е число от 1 до 5) в папка `XMLs`.
- `Incorrect-restaurantZ.xml` – 2 невалидни и лошо конструирани xml-файла (Z е число от 1 до 2) в папка `XMLs`..
- `restaurant_DOM.xml` - валиден xml, генериран чрез JavaScript в папка `XMLs`, чието съдържание отговаря на съдържанието на `restaurant4.xml`.
- `restaurant.js` - JavaScript файлът, който се използва за генерирането на `restaurant_DOM.xml`
- `dish-nutrition.xsl` , `recipes-info.xsl`, `restaurant-info.xsl` и `workers-info.xsl` - 4 xsl трансформации, които превръщат xml-файл в `dish-nutrition.html` , `recipes-info.html` , `restaurant-info.html` и `workers-info.html` . Трансформациите и резултатите са разположени в папка `XSLTs\toHTML`.
- `info_for_cooking.xsl` , `info_for_consumers.xsl` – 2 xsl трансформации, които превръщат даден xml-файл в `info_for_cooking.xml` и `info_for_consumers.xml`. Трансформациите и резултатите са разположени в папка `XSLTs\toXML`.
- `restaurant_staff.xsl` и `products_needed_for_a_recipe.xsl` – 2 xsl трансформации, които превръщат даден xml-файл в `restaurant_staff.txt` и `products_needed_for_a_recipe.txt` . Трансформациите и резултатите са разположени в папка `XSLTs\toText`.

Елементите, от които се състои описанието на ресторана са:
<!ELEMENT restaurant (restaurantName, description, address, telephone, staff, cookingSecrets) >

- Име на ресторант (*restaurantName*)
- Описание (*description*)
- Адрес (*address*)
- Телефон (*telephone*)
- Персонал (*staff*)
- Тайни от кухнята на ресторана (*cookingSecrets*)

Елемент ***restaurantName*** е от тип PCDATA.

Елемент ***description*** също е от тип PCDATA.

Елемент ***address*** се състои от елементи:

<!ELEMENT address (city, street, number) >

- ◆ *Град* (*city*) от тип PCDATA
- ◆ *Улица* (*street*) от тип PCDATA
 - Атрибут:
 - *type* от изброим тип, съставен от две възможни стойности – ул. | бул. , по подразбиране е ул.
- ◆ *Номер* (*number*) от тип PCDATA

Елемент ***telephone*** е от тип PCDATA.

Елемент **staff** се състои от елементи :

<!ELEMENT staff (chef,manager,assistantChef+,waiter+,cleaner+) >

◆ **Главен готовач (chef)** се състои от:

● Елементи:

- *chefFName* от тип PCDATA
- *chefLName* от тип PCDATA

● Атрибут:

- *заплата (salary)* , този атрибут задължително трябва да присъства за елемента *chef* в xml-файловете . Това в DTD се отбелязва така :

<!ATTLIST chef salary CDATA #REQUIRED >

◆ **Управител (manager)** състои от:

● Елементи:

- *managerFName* от тип PCDATA
- *managerLName* от тип PCDATA

● Атрибут:

- *заплата (salary)* , този атрибут задължително трябва да присъства за елемента *manager* в xml-файловете.

Това в DTD се отбелязва така :

<!ATTLIST manager salary CDATA #REQUIRED >

◆ **Помощник-готвач (assistantChef)** – в DTD е дефинирано с *assistantChef+*, защото трябва да има поне един такъв, но и е възможно да са повече в зависимост от нуждите на ресторанта. Този елемент се състои от :

● Елементи:

- *asschefFName* от тип PCDATA
- *asschefLName* от тип PCDATA

- Атрибут:

- *заплата (salary)* , този атрибут задължително трябва да присъства за елемента помощник-готвач (assistantChef) в xml-файловете. Това в DTD се отбелязва така :

```
<!ATTLIST assistantChef salary CDATA #REQUIRED >
```

- ◆ **Сервитьор (waiter)** - в DTD е дефинирано с waiter+, защото трябва да има поне един сервитьор, но може и повече. Този елемент се състои от :

- Елементи:

- *waiterFName* от тип PCDATA
- *waiterLName* от тип PCDATA

- Атрибут:

- *заплата (salary)* , този атрибут задължително трябва да присъства за елемента сервитьор (waiter) в xml-файловете. Това в DTD се отбелязва така :

```
<!ATTLIST waiter salary CDATA #REQUIRED >
```

- ◆ **Хигиенист (cleaner)** - в DTD е дефинирано с cleaner+, защото трябва да има поне един чистач, възможно е да са повече, когато ресторантът е с по-голям капацитет. Този елемент се състои от :

- Елементи:

- *cleanerFName* от тип PCDATA
- *cleanerLName* от тип PCDATA

- Атрибут:

- *заплата (salary)* , този атрибут задължително трябва да присъства за елемента хигиенист (cleaner) в xml-файловете. Това в DTD се отбелязва така :

```
<!ATTLIST cleaner salary CDATA #REQUIRED >
```

Елемент **тайни от кухнята на ресторантa (cookingSecrets)** се състои от : <!ELEMENT cookingSecrets (recipe+)>

◆ **Готварска рецепта (recipe)** . Тъй като всяко заведение за хранене трябва да има поне един специалитет и съответно рецепта за него, в DTD е отбелязано така <!ELEMENT cookingSecrets (recipe+)>. Елементът готварска рецепта има :

● Атрибути:

- *уникален номер на рецепта за дадено ястие (id)* , който задължително трябва да присъства за елемент рецепта, от тип ID е .
- *вегетарианско ли е ястието (vegetarian)* от изброим тип с две възможни стойности (да|не) , по подразбиране е “не”.
- *за кой сезон е подходящо ястието (season)* от изброим тип с четири възможни стойности (пролет|лято|есен|зима) , по подразбиране е “пролет”.
- *степен на трудност на рецептата (difficultRate)* от изброим тип с три възможни стойности (лесна|трудна|средна) , по подразбиране е “лесна”.
- *метод, по който се приготвя ястието (cookingMethod)* от изброим тип с възможни стойности (варене | задушаване | фурна | скара | барбекю | пържене | микровълнова | мариноване | консервиране | сухово), по подразбиране е “сухово”.

Тези атрибути за елемент рецепта са дефинирани по следния начин в DTD :

```
<!ATTLIST recipe id ID #REQUIRED
    vegetarian (да | не) "да"
    season (пролет | лято | есен | зима) "пролет"
```

difficultRate (лесна | средна | трудна) "лесна"
cookingMethod (варене | задушаване | фурна
| скара | пържене | микровълнова | мариноване |
консервиране | сухово) "сухово">

• Елементи:

<!ELEMENT recipe (dishName, ingredientList,
instructionList, advise*, nutrition, preparationTime,
cookingTime, appropriateDrink*)> .

- **име на ястиято (dishName)** от тип PCDATA.
<!ELEMENT dishName (#PCDATA) >
- **списък със съставки (ingredientList)**, състоящ
се от един или повече от един елемент **ingredient**.
<!ELEMENT ingredientList (ingredient+) >
Елемент ingredient се състои от:
 - елементи
<!ELEMENT ingredient (amount,unit,name) >
 - *количество (amount)* от тип PCDATA
 - *мерна единица (unit)* от тип PCDATA
 - *име на съставката (name)* от тип PCDATA
 - атрибут
<!ATTLIST ingredient optional (да | не) "не" >
 - по желание (optional) от изброям тип с две
възможни стойности (да|не). Стойността по
подразбиране е "не".
- **списък с указания за приготвяне
(instructionList)**, състоящ се от един или повече
от един елемент **step**, който е от тип PCDATA.
<!ELEMENT instructionList (step+) >
<!ELEMENT step (#PCDATA) >
- **съвет на готвача (advise)**, който може или да
присъства неограничен брой пъти, или да не

присъства нито веднъж (това е отбелязано със звездичка *). Той е от тип PCDATA.

- **информация за енергийната стойност в 1 порция (nutrition)** – този елемент се състои от :

- **елементи**

```
<!ELEMENT nutrition (energy, fat,  
carbohydrates, protein) >  
  
- енергийна стойност(energy) от тип  
PCDATA. Този елемент има атрибут in , който  
носи информация в каква мерна единица е  
измерена енергията – двата възможни  
варианта са ККал или КДжаули.  
<!ATTLIST energy in (ККал | КДжаули )  
"ККал">
```

- мазнини (fat) от тип PCDATA

- въглехидрати (carbohydrates) от тип
PCDATA

- белтъчини (protein) от тип PCDATA

- **необходимо време за подготовка (preparationTime)** от тип PCDATA.
<!ELEMENT preparationTime (#PCDATA)>
- **необходимо време за готовене (cookingTime)** от тип PCDATA.
<!ELEMENT cookingTime (#PCDATA) >
- **подходящо питие (appropriateDrink)** от тип PCDATA. Тъй като за някое ястие може да има
повече от 1 подходящо питие, а за друго може да
няма подходящо питие, този елемент може да се
среща о или много пъти в дадена рецепта (това е
отбелязано със звездичка *).

Така изградената йерархия се представя със следния Document Type Definition .

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT restaurant (restaurantName, description,
address,telephone,staff,cookingSecrets) >
<!ELEMENT restaurantName (#PCDATA) >
<!ELEMENT description (#PCDATA) >
<!ELEMENT address (city,street,number) >
<!ELEMENT city (#PCDATA) >
<!ELEMENT street (#PCDATA) >
    <!ATTLIST street type (ул. | бул.) "ул.">
<!ELEMENT number (#PCDATA) >
<!ELEMENT telephone (#PCDATA) >
<!ELEMENT staff (chef,manager,assistantChef+,waiter+,cleaner+) >
<!ELEMENT chef (chefFName,chefLName) >
    <!ATTLIST chef salary CDATA #REQUIRED >
<!ELEMENT chefFName (#PCDATA) >
<!ELEMENT chefLName (#PCDATA) >
<!ELEMENT manager (managerFName,managerLName) >
    <!ATTLIST manager salary CDATA #REQUIRED >
<!ELEMENT managerFName (#PCDATA) >
<!ELEMENT managerLName (#PCDATA) >
<!ELEMENT assistantChef (asschefFName,asschefLName) >
    <!ATTLIST assistantChef salary CDATA #REQUIRED >
<!ELEMENT asschefFName (#PCDATA) >
<!ELEMENT asschefLName (#PCDATA) >
<!ELEMENT waiter (waiterFName,waiterLName) >
    <!ATTLIST waiter salary CDATA #REQUIRED >
<!ELEMENT waiterFName (#PCDATA) >
<!ELEMENT waiterLName (#PCDATA) >
<!ELEMENT cleaner (cleanerFName,cleanerLName) >
    <!ATTLIST cleaner salary CDATA #REQUIRED >
<!ELEMENT cleanerFName (#PCDATA) >
<!ELEMENT cleanerLName (#PCDATA) >
<!ELEMENT cookingSecrets (recipe+) >
<!ELEMENT recipe
(dishName,ingredientList,instructionList,advise*,nutrition,preparationTime,cookingTime,appropriateDrink*)>
    <!ATTLIST recipe id ID #REQUIRED
        vegetarian (да | не) "не"
```

```

season (пролет | лято | есен | зима) "пролет"
difficultRate (лесна | средна | трудна) "лесна"
cookingMethod (варене | задушаване | фурна | скара | пържене |
микровълнова | мариноване | консервиране | сухово) "сухово">
<!ELEMENT dishName (#PCDATA) >
<!ELEMENT ingredientList (ingredient+) >
<!ELEMENT ingredient (amount,unit,name) >
  <!ATTLIST ingredient optional (да | не) "не" >
<!ELEMENT amount (#PCDATA)>
<!ELEMENT unit (#PCDATA) >
<!ELEMENT name (#PCDATA) >
<!ELEMENT instructionList (step+) >
<!ELEMENT step (#PCDATA) >
<!ELEMENT advise (#PCDATA) >
<!ELEMENT nutrition (energy,fat,carbohydrates,protein) >
<!ELEMENT energy (#PCDATA) >
  <!ATTLIST energy in (ККал | КДжаули ) "ККал">
<!ELEMENT fat (#PCDATA) >
<!ELEMENT carbohydrates (#PCDATA) >
<!ELEMENT protein (#PCDATA)>
<!ELEMENT preparationTime (#PCDATA)>
<!ELEMENT cookingTime (#PCDATA) >
<!ELEMENT appropriateDrink (#PCDATA) >

```

Създадени са 5 xml-файла, които са добре конструирани и валидни за тази Document Type Definition (restaurant1.xml , restaurant2.xml , restaurant3.xml , restaurant4.xml , restaurant5.xml) .

В края на документацията е приложено описание на грешките в лошо конструираните и невалидни за това DTD **xml-документи¹** (Incorrect-restaurant1.xml , Incorrect-restaurant2.xml) и 1 вариант на добре структуриран и валиден за това DTD **xmlдокумент²** (restaurant1.xml).

Xml-документът restaurant_DOM.xml се генерира от файлът restaurant.js , който е със следното съдържание:

```
var document= new ActiveXObject ("MSXML.DOMDocument");
//Създава се root-елемента на документа
var restaurant = document.createElement('restaurant');
document.appendChild(restaurant);
//Създават се преките наследници на root-елемента
// 1. Име на ресторант
addElem(restaurant,'restaurantName','Лебед');
// 2. Описание на ресторант
addElem(restaurant,'description','Отлична кухня с традиционно меню и
голямо разнообразие на морски деликатеси и риба');
// 3. Адрес
setAddress (restaurant,' София','бул. ',' Самоковско шосе','83');
// 4. Телефон
addElem(restaurant,'telephone','992 11 11');
// 5. Служители
var staff=document.createElement('staff');
    addChef(staff,'Михайл','Зидаров','1400');
    addManager(staff,'Златин','Димитров','1000');
    addAssistantChef(staff,'Надежда','Иванова','800');
    addAssistantChef(staff,'Светослава','Трайкова','740');
    addWaiter(staff,'Мария','Василова','650');
    addWaiter(staff,'Мирослава','Трендафилова','620');
    addCleaner(staff,'Благовеста','Димитрова','600');
restaurant.appendChild(staff);
// 6. Тайни от кухнята
var cookingSecrets=document.createElement('cookingSecrets');
----- Първа рецепта -----
var recipe=document.createElement('recipe');
setDishName(recipe,' Паста със съомга');
setRecipeAttributes(recipe,'T001','не','есен','средна','варене');

var ingredientList=document.createElement('ingredientList');
    addIngredient(ingredientList,'600','грама','талятели, фетучини
или спагети','не');
    addIngredient(ingredientList,'4',' с.л. ','зехтин ','не');
    addIngredient(ingredientList,'400 ','грама ','филе от съомга,
съомгова пъстърва, пангасиус, тилапия или др. ','не');
    addIngredient(ingredientList,'3 ','скилидки ','чесън','не');
    addIngredient(ingredientList,'100','грама','спанак','не');
    addIngredient(ingredientList,'1/2','брой','зелена лята чушка,
нарязана на ситно','не');
    addIngredient(ingredientList,'2','броя','печени чушки, нарязани
на лентички ','не');
    addIngredient(ingredientList,'1 ','с.л. ','балсамов оцет','не');
    addIngredient(ingredientList,'1 ','ч.л. ',' лимонова кора','не');
    addIngredient(ingredientList,'1 ','щипка ',' сол ','да');
    addIngredient(ingredientList,'1 ','щипка ',' пипер ','да'));
recipe.appendChild(ingredientList);

var instructionList=document.createElement('instructionList');
    setInstructionStep(instructionList,'Пастата се сварява според
инструкциите на опаковката. Прецежда се.');
```

```

        setInstructionStep(instructionList, 'Рибата се посолява и се
поръсва с пипер. Намазва се с 1 с. л. от зехтина и се пече от двете
страни в загрят тиган. Ако парчето филе е цяло, се пече по около 3 мин.
от всяка страна. Оставя се да поизстине и се накъсва на парченца. ');
        setInstructionStep(instructionList, 'В останалия загрят зехтин се
слагат чесънът и лютата чушка и се разбъркват много кратко. Добавят се
спанакът и печените чушки. Разбъркват се около минута. Добавя се
балсамовият оцет и лимоновата кора. ');
        setInstructionStep(instructionList, 'Пастата се разбърква със
смesta от чушки и спанак и се гарнира с рибата. ');
recipe.appendChild(instructionList);

addNutritionInfo(recipe, '320', 'ККал', '20', '60', '30');
setPrepTime(recipe, '20');
setCookTime(recipe, '30');
setApprDrink(recipe, 'бяло вино');
cookingSecrets.appendChild(recipe);
//---- Втора рецепта -----
recipe=document.createElement('recipe');
setDishName(recipe, ' Кюфтенца с царевица и риба тон ');
setRecipeAttributes(recipe, 'T002', 'не', 'лято', 'средна', 'пържене');

ingredientList=document.createElement('ingredientList');
        addIngredient(ingredientList,'1 ','ч.ч.', ' царевични зърна,
сварени и отцедени','');
        addIngredient(ingredientList,'200', ' грама ', 'консервирана риба
тон','');
        addIngredient(ingredientList, ' 200', 'грама ', 'сварени
картофи','');
        addIngredient(ingredientList, '1 ', 'брой', 'яйце','');
        addIngredient(ingredientList, '1 ', 'ч.л.', 'листа от прясна
машерка','');
        addIngredient(ingredientList, '2 ', 'с.л.', 'нарязан зелен лук ');
        addIngredient(ingredientList, '1 ', 'скилидка', 'ситно накълцан
чесън');
        addIngredient(ingredientList, '1 ', 'с.л.', 'лимонов сок');
        addIngredient(ingredientList, '1 ', 'ч.л.', 'настъргана на ситно
лимонова кора');
        addIngredient(ingredientList, '1 ', 'ч.л. ', 'лют сос ');
        addIngredient(ingredientList, '2 ', 'с.л', ' брашно с набухватели
','');
        addIngredient(ingredientList, '2 ', 'с.л', ' бяло брашно за
овалване');
        addIngredient(ingredientList, '2 ', 'шипки', 'сол', 'да');
        addIngredient(ingredientList, '1 ', 'шипка', 'пипер ', 'да');
        addIngredient(ingredientList, '4 ', 'с.л', 'олио за пържене');
recipe.appendChild(ingredientList);

instructionList=document.createElement('instructionList');
        setInstructionStep(instructionList, 'Царевицата се смесва с
отцедените парченца риба, намачканите картофи, яйцето, кимионът,
машерката, лукът, чесънът, лимоновият сок, лимоновата кора, лютият сос
и брашното. ');
        setInstructionStep(instructionList, 'Оформят се кюфтенца, които се
овалват в брашно и се изпържват в сгорещеното олио от двете страни. ');
recipe.appendChild(instructionList);
setAdvise(recipe, ' Кюфтенцата се сервираат със свежа салата. ')

```

```

addNutritionInfo(recipe,'320','ККал','28','35','50');
setPrepTime(recipe,'30');
setCookTime(recipe,'20');
setApprDrink(recipe,'чаша студена бира');
cookingSecrets.appendChild(recipe);

restaurant.appendChild(cookingSecrets);
document.save('restaurant_DOM.xml');
// ----- ФУНКЦИИ -----
function
addIngredient(ingredientList,i_amount,i_unit,i_name,is_optional)
{
    var ingredient=document.createElement('ingredient');
    switch(is_optional)
    {
        case 'да' :
            ingredient.setAttribute('optional',is_optional);
            break;
        default: ingredient.setAttribute('optional','не');
    }
    addElem(ingredient,'amount',i_amount);
    addElem(ingredient,'unit',i_unit);
    addElem(ingredient,'name',i_name);
    ingredientList.appendChild(ingredient);
}
function setInstructionStep(instructionList,i_step)
{
    var step=document.createElement('step');
    var textNode = document.createTextNode(i_step);
    step.appendChild(textNode);
    instructionList.appendChild(step);
}
function setPrepTime(recipe,prepTime)
{
    addElem(recipe,'preparationTime',prepTime);
}
function setCookTime(recipe,cookTime)
{
    addElem(recipe,'cookingTime',cookTime);
}
function setAdvise(recipe,ad)
{
    addElem(recipe,'advise',ad);
}
function setApprDrink(recipe,drink)
{
    addElem(recipe,'appropriateDrink',drink)
}
function setDishName(recipe,d_name)
{
    addElem(recipe,'dishName',d_name);
}

```

```

function
addNutritionInfo(recipe,r_energy,r_in,r_fat,r_carbohydrates,r_protein)
{
    var nutrition=document.createElement('nutrition');
    switch(r_in)
    {
        case 'КДжаули':
            addElem(nutrition,'energy',r_energy,'in',r_in);
            break;
        default:
            addElem(nutrition,'energy',r_energy,'in','ККал');
    };
    addElem(nutrition,'fat',r_fat);
    addElem(nutrition,'carbohydrates',r_carbohydrates);
    addElem(nutrition,'protein',r_protein);
    recipe.appendChild(nutrition);
}
function
setRecipeAttributes(recipe,r_id,r_vegetarian,r_season,r_difficultRate,r_cookingMethod)
{
    if(r_id)
    {
        recipe.setAttribute('id',r_id);
        switch (r_vegetarian){
            case 'да':
                recipe.setAttribute('vegetarian',r_vegetarian); break;
            default: recipe.setAttribute('vegetarian','не');
        };
        switch (r_season){
            case 'лято': recipe.setAttribute('season',r_season);
            break;
            case 'есен' : recipe.setAttribute('season',r_season);
            break;
            case 'есен' : recipe.setAttribute('season',r_season);
            break;
            default: recipe.setAttribute('season','пролет');
        };
        switch (r_difficultRate){
            case 'средна':
                recipe.setAttribute('difficultRate',r_difficultRate); break;
            case 'трудна' :
                recipe.setAttribute('difficultRate',r_difficultRate); break;
            default:
                recipe.setAttribute('difficultRate','лесна');
        };
        switch (r_cookingMethod){
            case 'варене':
                recipe.setAttribute('cookingMethod',r_cookingMethod); break;
            case 'задушаване':
                recipe.setAttribute('cookingMethod',r_cookingMethod); break;
            case 'форна':
                recipe.setAttribute('cookingMethod',r_cookingMethod); break;
            case 'скара':
                recipe.setAttribute('cookingMethod',r_cookingMethod); break;
            case 'пържене':
                recipe.setAttribute('cookingMethod',r_cookingMethod); break;
        };
    };
}

```

```

        case 'микровълнова':
    recipe.setAttribute('cookingMethod', r_cookingMethod); break;
        case 'мариноване':
    recipe.setAttribute('cookingMethod', r_cookingMethod); break;
        case 'консервиране':
    recipe.setAttribute('cookingMethod', r_cookingMethod); break;
    default:
    recipe.setAttribute('cookingMethod', 'сухово');
}
}

function addChef(staff, fname, lname, s)
{
    var chef=document.createElement('chef');
    chef.setAttribute('salary',s);
    addElem(chef,'chefFName', fname);
    addElem(chef,'chefLName', lname);
    staff.appendChild(chef);
}

function addManager(staff, fname, lname, s)
{
    var manager=document.createElement('manager');
    manager.setAttribute('salary',s);
    addElem(manager,'managerFName', fname);
    addElem(manager,'managerLName', lname);
    staff.appendChild(manager);
}

function addAssistantChef(staff, fname, lname, s)
{
    var assistantChef=document.createElement('assistantChef');
    assistantChef.setAttribute('salary',s);
    addElem(assistantChef,'asschefFName', fname);
    addElem(assistantChef,'asschefLName', lname);
    staff.appendChild(assistantChef);
}

function addWaiter(staff, fname, lname, s)
{
    var waiter=document.createElement('waiter');
    waiter.setAttribute('salary',s);
    addElem(waiter,'waiterFName', fname);
    addElem(waiter,'waiterLName', lname);
    staff.appendChild(waiter);
}

function addCleaner(staff, fname, lname, s)
{
    var cleaner=document.createElement('cleaner');
    cleaner.setAttribute('salary',s);
    addElem(cleaner,'cleanerFName', fname);
    addElem(cleaner,'cleanerLName', lname);
    staff.appendChild(cleaner);
}

function setAddress (restaurant,r_city,r_type,r_street,r_number)
{
    var address=document.createElement('address');
    addElem(address,'city',r_city);
    switch(r_type)

```

```

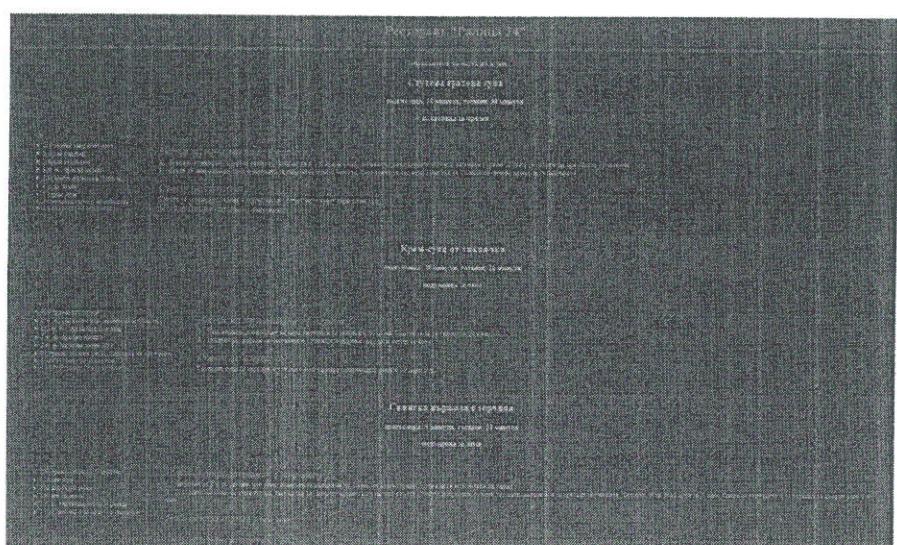
    {
    case 'бул.':
        addElem(address, 'street', r_street, 'type', r_type);
        break;
    default:
        addElem(address, 'street', r_street, 'type', 'ул. ');
    };
    addElem(address, 'number', r_number);
    restaurant.appendChild(address);
}

function addElem(node, elementName, data, attributeName, attributeData)
{
    if(data)
    {
        var element =
document.createElement(elementName);
        node.appendChild(element);
        if (attributeData)
        {
            element.setAttribute(attributeName,
attributeData);
        }
        var textNode = document.createTextNode(data);
        element.appendChild(textNode);
    }
}

```

В проекта са включени и 4 различни стилови таблици за трансформиране на XML документите в html-формат (приложени в папката XSLTs\toHTML заедно с получените резултати) , 2 различни стилови таблици за трансформиране на XML документите в други XML документи (приложени в папката XSLTs\toXML заедно с получените резултати) и 2 различни стилови таблици за трансформиране на XML документите в текстов формат (приложени в папката XSLTs\toTEXT заедно с резултатите) .

Представеното по-долу XSLT(recipes-info.xsl) визуализира в html рецептите от даден xml-документ в следния формат :



```

<?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" version="1.0" encoding="UTF-8" />
  <xsl:template name="header">
    <hr size="6" />
    <h1 align="center" > Ресторант "<xsl:value-of select="//restaurantName"/>"</h1>
    <hr size="6" />
    <h4 align="center" > <xsl:value-of select="//description"/></h4>
  </xsl:template>
  <xsl:template name="recipe">
    <xsl:for-each select="//recipe">
      <div style=" background-color:#5e0300;">
        <h2 align="center" style="color:#f6f2f3;font:serif;">
          <xsl:value-of select=".//dishName" />
        </h2>
        <h4 align="center" style="color:#f6f2f3;font:serif;">
          подготвка: <xsl:value-of select=".//preparationTime"/> минути,
          готовене: <xsl:value-of select=".//cookingTime"/> минути</h4>
        <h4 align="center" style="color:#f6f2f3;font:serif;"> подходяща за <xsl:value-of
        select=".//@season"/></h4>
        <table cellpadding="30px">
          <tr>
            <td nowrap="nowrap">
              <ul>
                <xsl:for-each select=".// ingredientList/ingredient">
                  <xsl:if test=".//@optional='да'">
                    <li style="font: italic 15px serif; color: #b38154; "> <xsl:value-of
                    select="concat(.//amount,.//unit,.//name)"/> ( по желание )</li>
                  </xsl:if>
                  <xsl:if test=".//@optional!='да'">
                    <li><xsl:value-of select="concat(.//amount,.//unit,.//name)"/></li>
                  </xsl:if>
                </xsl:for-each>
              </ul>
            </td>
            <td> <span style=" color: #b38154; font: italic 25px serif; ">Начин на
              приготвяне : [ <xsl:value-of select=".//cookingMethod"/> ] <br/> </span>
              <xsl:for-each select=".//instructionList/step">
                <xsl:value-of select="position()"/> .
                <xsl:value-of select="text()"/>
              </xsl:for-each>
            </td>
          </tr>
        </table>
      </div>
    </xsl:for-each>
  </xsl:template>

```

```

<br/>
</xsl:for-each>
<xsl:if test=".//advise">
<br/>
<span style=" color: #b38154; font: italic 25px serif;"> Съвет на готвача
: <br/></span>
<xsl:value-of select=".//advise"/>
</xsl:if>
<xsl:if test=".//appropriateDrink">
<br/>
<span style=" color: #b38154; font: italic 25px serif;"> Подходящо питие
: </span>
<xsl:value-of select=".//appropriateDrink"/>
</xsl:if>
</td>
</tr>
</table>
</div>
</xsl:for-each>
</xsl:template>
<xsl:template match='/'>
<html>
<head>
<title> Ресторант "<xsl:value-of select="//restaurantName"/>" </title>
</head>
<body bgcolor="#870000" text="#F9ab3e">
<xsl:call-template name="header"></xsl:call-template>
<xsl:call-template name="recipe"></xsl:call-template>
</body>
</html>
</xsl:template>
</xsl:stylesheet>

```

В него са създадени 3 шаблона :

- `<xsl:template match="/">` се изпълнява при срещане на root-елемента в xml-документа. С него се създават html,head,title и body тага за html-документа.
`<title> Ресторант "<xsl:value-of select="//restaurantName"/>" </title>` създава title на страницата като взима името на ресторана (restaurantName), което е единствено в xml-файла. В body-тага се извикват останалите два шаблона с имена header и recipe.
- `<xsl:template name="header">` Визуализира по красив начин името на ресторана оградено отгоре и отдолу с хоризонтална линия . Под него с по-малък шрифт се изписва описание на ресторана. Този шаблон се изпълнява само веднъж.
- `<xsl:template name="recipe">` се изпълнява за всеки срещнат елемент recipe в xml документа. За всяка рецепта се създава div-таг, в който се помества информацията за рецептата в оцветен в кафяво блок. Като заглавие са оформени името на ястието, времето необходимо за подготовката и готовното, както и за кой сезон е подходящо,ако не е посочен сезон се извежда сезонът по подразбиране -пролет. След това се създава таблица с 1 ред, в който има 2 клетки. В първата клетка се записват съставките, а във втората начинът на приготвяне, съвет на готвача(ако има такъв в xml-файла) и подходящо питие (ако има такова в xml-файла). Проверката са наличие/отсъствие на съвет за текущата рецепта е осъществена така: (за питие аналогично)

```
<xsl:if test=".//advise">
    <br/>
    <span style=" color: #b38154; font: italic 25px serif;"> Съвет на
готвача : </span>
    <xsl:value-of select=".//advise"/>
</xsl:if>
```

Съставянето на неномерираания списък с продуктите за дадена рецепта следва следната логика – ако продуктът не е задължителен (тоест стойността на атриутът optional е “да”) се съставя елемент на списъка с различно форматиране- по-тъмен цвят и наклонен шрифт последвано от “(по желание)”;

```
<xsl:if test="./@optional='да'">
    <li style="font: italic 15px serif; color: #b38154; "> <xsl:value-of
select="concat(./amount,./unit,./name)" /> ( по желание )</li>
</xsl:if>
```

ако продуктът е задължителен (тоест стойността на атриутът optional е “не”, това е стойността по подразбиране) се съставя елемент на списъка с нормално форматиране.

```
<xsl:if test="./@optional!='да'">
    <li><xsl:value-of select="concat(./amount,./unit,./name)" /></li>
</xsl:if>
```

Начинът на приготвяне включва информация за вида на готовенето и стъпките, през който трябва да се премине. Всяка стъпка се номерира като се използва функцията `position()`.

```
<xsl:for-each select=".//instructionList/step">
    <xsl:value-of select="position()"/> .
    <xsl:value-of select="text()"/>
    <br/>
</xsl:for-each>
```

XML-документът *Incorrect-restaurant1.xml* :

- не е добре конструиран, защото таговете на елементите cookingTime и preparationTime се кръстосват.

```
<preparationTime>5<cookingTime>
</preparationTime> 20</cookingTime>
```

Правилно е `<preparationTime>5</preparationTime>`
`<cookingTime>20</cookingTime>`

- не е добре конструиран, защото в него има два root-елемента restaurant. Правилно е да бъде само един.

```
<?xml version="1.0" encoding='UTF-8' ?>
<restaurant> ....</restaurant>
<restaurant> ....</restaurant>
```

- не е добре конструиран, защото стойността на един атрибут не е в кавички `<energy in=KKал>400</energy>`.

Правилно е `<energy in="KKал">400</energy>`

- не е добре конструиран, защото един атрибут е употребен повече от веднъж, но с различна стойност в един и същ елемент. Правилно е само веднъж да се използва.

```
<recipe id="R001" vegetarian="не" season="пролет" season="есен"
difficultRate="лесна" cookingMethod="варене" >
```

- не е валиден за цитираното DTD, защото стойността на атрибута in на елемент energy - `<ingredient optional="nein" >`, не е нито една от двете посочени възможни стойности в DTD.

```
<!ATTLIST ingredient optional (да | не) "не" >
```

- не е валиден за цитираното DTD, защото липсва заплата за главния готвач `<chef>`

```
<chefFName> Стефан </chefFName>
<chefLName> Николов </chefLName>
</chef>
```

А в DTD е посочено, че този атрибут е задължителен.

```
<!ATTLIST chef salary CDATA #REQUIRED >
```

XML-документът *Incorrect-restaurant2.xml* :

- не е валиден за цитираното DTD, защото липсва елемент telephone. А в DTD е посочено, че този елемент задължително трябва да пръсъства веднъж.

```
<!ELEMENT restaurant (restaurantName, description, address, telephone, staff, cookingSecrets) >
```

- не е валиден за цитираното DTD, защото има двама главни готвачи.

```
<chef salary="1100" >
  <chefFName> Вяра </chefFName>
  <chefLName> Илиева </chefLName>
</chef>
<chef salary="700" >
  <chefFName> Милко</chefFName>
  <chefLName>Захариев </chefLName>
</chef>
```

DTD разрешава само един :

```
<!ELEMENT staff (chef, manager, assistantChef+, waiter+, cleaner+) >
```

- не е валиден за цитираното DTD, защото в ресторанта няма чистач, а според DTD е задължително да има 1 или повече.

```
<!ELEMENT staff (chef, manager, assistantChef+, waiter+, cleaner+) >
```

- не е добре конструиран, защото стойността на атриут type не е оградена от едни и същи кавички от двете страни.

```
<street type='бул.'> Христо Ботев</street>
```

Правилно е `<street type="бул."> Христо Ботев</street>` или

```
<street type='бул.'> Христо Ботев</street>
```

² Добре конструиран и валиден спрямо цитираната DTD xml-документ restaurant1.xml .

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE restaurant SYSTEM "restaurant.dtd">
<restaurant>
  <restaurantName>Ралица 34</restaurantName>
  <description> Специалитети по вкуса на всеки </description>
  <address>
    <city> София </city>
    <street type=" ул. "> Ралица</street>
    <number> 34 </number>
  </address>
  <telephone> 988 22 21 </telephone>
  <staff>
    <chef salary="800">
      <chefFName> Стефан </chefFName>
      <chefLName> Николов </chefLName>
    </chef>
    <manager salary="600">
      <managerFName> Петър </managerFName>
      <managerLName> Радимиров </managerLName>
    </manager>
    <assistantChef salary="450">
      <asschefFName> Мария </asschefFName>
      <asschefLName> Георгиева </asschefLName>
    </assistantChef>
    <assistantChef salary="450">
      <asschefFName> Павлина </asschefFName>
      <asschefLName> Димитрова </asschefLName>
    </assistantChef>
    <waiter salary ="400">
      <waiterFName> Наталия </waiterFName>
      <waiterLName> Ушева </waiterLName>
    </waiter>
    <waiter salary="400">
      <waiterFName> Цветелина </waiterFName>
      <waiterLName> Михайлова </waiterLName>
    </waiter>
```

```
<waiter salary="400">
    <waiterFName> Добромира </waiterFName>
    <waiterLName> Цветанова </waiterLName>
</waiter>
<cleaner salary="250">
    <cleanerFName> Здравка </cleanerFName>
    <cleanerLName> Петрова </cleanerLName>
</cleaner>
</staff>
<cookingSecrets>
    <recipe id="R001" vegetarian="не" season="пролет" difficultRate="лесна"
cookingMethod="варене" >
        <dishName> Студена грахова супа </dishName>
        <ingredientList>
            <ingredient optional="не" >
                <amount> 250 </amount>
                <unit> грама </unit>
                <name> замразен грах </name>
            </ingredient>
            <ingredient optional="не">
                <amount> 1 </amount>
                <unit> брой </unit>
                <name> картоф </name>
            </ingredient>
            <ingredient optional="не">
                <amount> 2 </amount>
                <unit> броя </unit>
                <name> моркови </name>
            </ingredient>
            <ingredient optional="не">
                <amount> 1 </amount>
                <unit> ч. ч. </unit>
                <name> прясно мляко </name>
            </ingredient>
            <ingredient optional="не">
                <amount> 2 </amount>
                <unit> кубчета </unit>
                <name> пилешки бульон </name>
            </ingredient>
```

```
<ingredient optional="не">
    <amount> 4 </amount>
    <unit> с. л. </unit>
    <name> олио </name>
</ingredient>
<ingredient optional="не">
    <amount> 1 </amount>
    <unit> стрък </unit>
    <name> лук </name>
</ingredient>
<ingredient optional="да">
    <amount> 1 </amount>
    <unit> скилидка </unit>
    <name> чесън </name>
</ingredient>
<ingredient optional="да">
    <amount> 1 </amount>
    <unit> ч. л. </unit>
    <name> къри </name>
</ingredient>
</ingredientList>
<instructionList>
    <step> Всички зеленчуци се режат на едро . Заливат се с 1 л вряла вода, добавят се натрошени кубчета пилешки бульон .Супата се оставя да ври около 15 минути. </step>
    <step> Слага се замразения грах и варенето продължава още 3 минути, след което се пюрира. Прибавя се кипналото прясно мляко и се разбърква. </step>
</instructionList>
<advise> Супата се сервира охладена и поръсена със ситно нарязаните пера зелен лук. </advise>
<nutrition>
    <energy in="ККал">160 </energy>
    <fat>5 </fat>
    <carbohydrates> 60</carbohydrates>
    <protein>30 </protein>
</nutrition>
<preparationTime> 10 </preparationTime>
<cookingTime> 30 </cookingTime>
<appropriateDrink> бяло вино</appropriateDrink>
```

```
</recipe>
<recipe id="R002" vegetarian="не" season="зима" difficultRate="средна"
cookingMethod="пържене">
    <dishName> Свинска пържола с горчица </dishName>
    <ingredientList>
        <ingredient optional="не">
            <amount>300 </amount>
            <unit> грама </unit>
            <name>свинско бонфиле</name>
        </ingredient>
        <ingredient optional="не">
            <amount>1 </amount>
            <unit> глава </unit>
            <name>лук</name>
        </ingredient>
        <ingredient optional="не">
            <amount>100 </amount>
            <unit> мл </unit>
            <name>бяло вино</name>
        </ingredient>
        <ingredient optional="не">
            <amount>100</amount>
            <unit> мл </unit>
            <name>сметана</name>
        </ingredient>
        <ingredient optional="не">
            <amount>1</amount>
            <unit> с. л. </unit>
            <name>пълнозърнеста горчица</name>
        </ingredient>
        <ingredient optional="не">
            <amount>2 </amount>
            <unit>с.л. </unit>
            <name> зехтин или олио за пържене</name>
        </ingredient>
    </ingredientList>
    <instructionList>
        <step>Загряват се 2 с. л. зехтин или олио и месото се запържва за 2 мин. от
        двете страни. Изважда се и се оставя на топло.
    </instructionList>
</recipe>
```

</step>

<step> В същата мазнина се слага лукът, разбърква се, докато омекне и се налива виното. Когато изври наполовина, се слага сметаната и се връщат пържолите. Оставят се на слаб огън за 8 мин. Слага се горчицата и се оставят да къкрят още 3-4 мин. </step>

</instructionList>

<nutrition>

<energy in="ККал" >400 </energy>

<fat> 10 </fat>

<carbohydrates> 19.9 </carbohydrates>

<protein> 21.5 </protein>

</nutrition>

<preparationTime>5</preparationTime>

<cookingTime>20</cookingTime>

<appropriateDrink> бяло вино</appropriateDrink>

</recipe>

</cookingSecrets>

</restaurant>