**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ГАБРОВО**

Катедра „Компютърни системи и технологии“

**КУРСОВА РАБОТА**

**Тема**: Напишете HTML5 страница, в която се визуализира текущото време във формат „час:минути“. Информацията за времето се получава от Web ресурс getCurrentTime чрез използване на AJAX заявки.

**Разработил**: Богомил Георгиев Иванов

**Фак. №**: 22372126 **Курс**: 2-ри курс **Специалност**: СКИ

**Проверил**:.......................................

**Подпис**:........................... **Дата**:........................

**Теоретична част:**

**Асинхронно и синхронно изпълнение на програмен код**

*Синхронно изпълнение*

Синхронното изпълнение означава, че задачите се изпълняват една след друга. Всяка следваща задача започва, след като предходната завърши. Това е стандартният начин на изпълнение на код в повечето програмни езици.

**Предимства:**

* Лесен за разбиране и отстраняване на грешки код.
* Логическата последователност на задачите е ясна.

**Недостатъци:**

* Ако една задача отнема много време (напр. мрежова заявка), програмата замръзва, докато задачата се изпълни.
* Намалена производителност, особено при обработка на множество задачи.

*Асинхронно изпълнение*

Асинхронното изпълнение позволява на програмата да изпълнява задачи, без да изчаква завършването на други. Това е особено полезно за операции с вход/изход, мрежови заявки и таймери.

**Предимства:**

* По-висока производителност, тъй като програмата може да обработва други задачи, докато чака за резултати от бавни операции.
* Подходящо за потребителски интерфейси, където блокирането е неприемливо.

**Недостатъци:**

* По-сложен за разбиране и поддръжка код.
* Управлението на зависимости между задачите изисква специални техники (напр. callback функции, промиси, async/await).

**Asynchronous JavaScript and XML (AJAX). Генериране на AJAX заявки чрез библиотеката jQuery**

AJAX е техника, която позволява на уеб страници да обменят данни със сървъра асинхронно, без да презареждат цялата страница. Това подобрява потребителското изживяване, като осигурява бърза и динамична реакция на уеб приложенията.

**Примерни приложения на AJAX:**

* Автоматично обновяване на съдържание (напр. времето, новини, курсове на валути).
* Динамично зареждане на части от страница (напр. зареждане на допълнителни коментари или продукти).
* Проверка на потребителски данни (напр. уникалност на имейл при регистрация).

**Предимства на използване на jQuery за AJAX**

* Оптимизиран и лесен за използване синтаксис.
* Поддържа различни браузъри, като се грижи за несъвместимостите.
* Предлага удобни методи като $.ajax(), $.get(), и $.post() за различни типове заявки.

**Програмна реализация:**

* *index.html*

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="utf-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>GetTimePrjct</title>**

**<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>**

**<script src="getTime.js"></script>**

**<link rel="stylesheet" href="style.css">**

**<link rel="preconnect" href="https://fonts.googleapis.com">**

**<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>**

**<link href="https://fonts.googleapis.com/css2?family=Nunito:ital,wght@0,200..1000;1,200..1000&display=swap" rel="stylesheet">**

**</head>**

**<body>**

**<h1 id="Title">Date and Time</h1>**

**<p id="time" class="time">Fetching time...</p>**

**<p id="date" class="date">Fetching date...</p>**

**</body>**

**<footer>**

**<p id="copy">&copy; Bogomil Ivanov 2024. Used for TU-Gab Assigments.</p>**

**</footer>**

**</html>**

* *getTime.js*

**let previousValue = null;**

**function getTimeASync () {**

**$.ajax({**

**url: "https://kst.tugab.bg/pis/gettimejson.php",**

**type: "GET",**

**success: function(response) {**

**console.info(response);**

**const currentValue = response.value;**

**if (currentValue !== previousValue) {**

**const [timeText, dateText] = response.value.split(", ");**

**const [hours, minutes] = timeText.split(":");**

**$("#time").text(`${hours}:${minutes}`);**

**$("#date").text(dateText);**

**previousValue = currentValue;**

**setTimeout(getTimeASync, 1000);**

**};**

**},**

**error: function(xhr, status, error) {**

**console.error("Failed to fetch data. Status:", xhr.status, "Error:", error);**

**setTimeout(getTimeASync, 5000);**

**}**

**});**

**}**

**getTimeASync();**

* *style.css*

**\* {**

**text-align: center;**

**font-family: "Nunito", serif;**

**font-optical-sizing: auto;**

**}**

**#Title {**

**font-size: 48px;**

**font-weight: 900;**

**}**

**#time, #date {**

**font-size: 24px;**

**font-weight: 500;**

**}**

**footer {**

**position: fixed;**

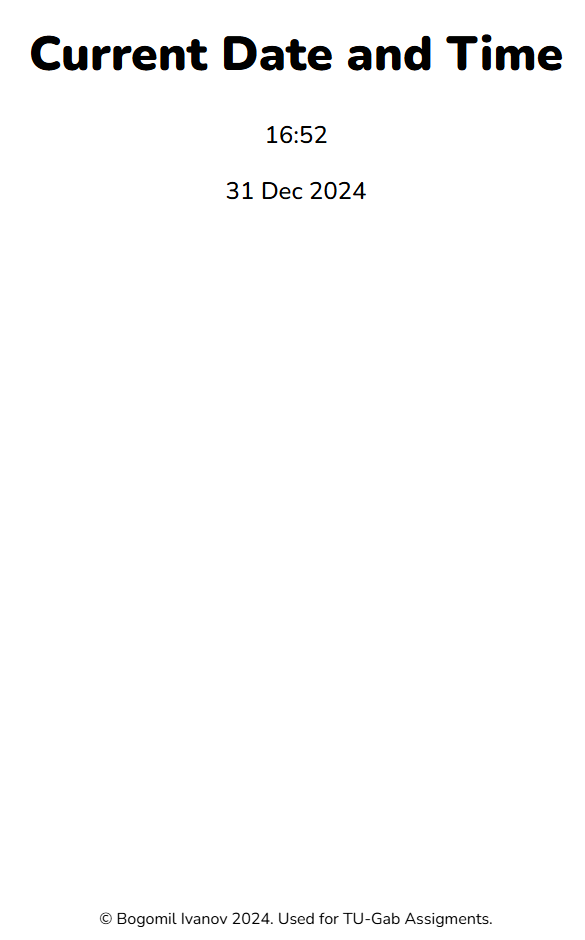
**left: 0;**

**bottom: 0;**

**width: 100%;**

**}**

**Резултат от програмния код:**



**Изводи:**

Използването на AJAX позволява по-добро потребителско изживяване чрез асинхронно зареждане на данни. Библиотеката jQuery значително опростява работата с AJAX заявки, като осигурява по-кратък и разбираем синтаксис. Основен недостатък на AJAX приложенията е зависимостта от стабилността и наличността на мрежовата връзка.

**Използвана литература:**

1. Mozilla Developer Network (MDN): AJAX Introduction
2. jQuery Documentation: jQuery.ajax()
3. David Flanagan, "JavaScript: The Definitive Guide", O'Reilly Media.