Szegedi Tudományegyetem

Informatikai Intézet

SZAKDOLGOZAT

Pál Krisztián Zoltán

2023

Szegedi Tudományegyetem

Informatikai Intézet

Italárusító webshop alkalmazás beépített játékkal

Szakdolgozat

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *Készítette:* |  | *Témavezető:* |  |
|  | Pál Krisztián Zoltán |  | Dr. Bilicki Vilmos |  |
|  | programtervező informatikus szakos hallgató |  | Oktató és konzulens |  |

Szeged

2023

# ***Feladatkiírás***

A szakdolgozat I. és szakdolgozat II. tárgyak keretein belül egy különböző alkoholos és alkoholmentes italokat árusító webáruházat alkot meg a hallgató, amelybe az általánosabb webshopokba található funkciók mellett egyedi funkciók is meg lesznek valósítva. Többek között beleértve a fizetési rendszer integrálását, emellett az árucikkek közötti gördülékeny böngészést.

A szakdolgozat teljesen öncélú, viszont felhasználhatósága kiterjedhet akár más sajátosan megalkotott, de ettől eltérő árucikkekkel foglalkozó weboldalak fejlesztése alapjaként. Emellett a megalkotott webalkalmazás meg fogja állni a helyét az interneten fellelhető más webshopokkal szemben.

Amióta az internet egyre elterjedtebbé vált, egyre népszerűbbek lettek a webshopok, melyek segítségével a felhasználók saját igényük szerint gyorsan tudnak böngészni a termékek tárházában. Mindig szükség van, és a jövőben is szükség lesz ezen platformokra, melyek megkönnyítik az emberek életét. Emellett a tulajdonosok számára is előny, hiszen rengeteg felhasználóhoz el tudják juttatni termékeiket.

A hallgató feladatát képezi egy újabb webshop megalkotása, mely azt az egyediséget viseli, hogy egy kis játék alkalmazás legyen beépítve a webshop keretein belülre, aminek segítségével különböző akciókra tehet szert a felhasználó. Így egyedi módon juttatjuk a felhasználót pénzügyi előnyhöz, emellett a webshopon is több időt tölt el, ami azt eredményezheti, hogy új termékeket ismer meg. Ezen üzleti logika elősegítheti, hogy a felhasználók több mindent rendeljenek, de még is szimpatikussá tegyük számukra a rendszert.

# ***Tartalmi összefoglaló***

* **A téma megnevezése:**

Italárusító webshop alkalmazás beépített játékkal

* **A megadott feladat megfogalmazása:**

A feladat egy olyan webalkalmazás fejlesztése, amely dinamikusan tölt be egy általános webshopot, melyen italok vásárolhatóak (természetesen csak virtuálisan). A webáruház kényelmes böngészést biztosít a felhasználónak a neki tetsző termékek között. Különlegessége a beépített játék, mely a felhasználóknak lehetőséget biztosít minden nap bizonyos kedvezmények megszerzéséhez.

* **A megoldási mód:**

Első tekintetben meg kellett ismernem az Angular keretrendszert körültekintőbben. Ezután komponensekbe, sőt modulokba szerveztem az alkalmazásom, ami eleinte csak egy játék nélküli webshop lesz. Szolgáltatásokat, materialokat és az Angular egyéb segítséget nyújtó eszközeit alkalmaztam a felhasználóbarát megjelenés végett. A TypeScript kódból könnyedén kapcsolódni tudtam a Firebase-hez, ami a Backendjét szolgáltatja a projektnek. Mindez után belemélyedek a webes játékfejlesztésbe, hogy be tudjam építeni a Super Mario-hoz hasonló játékot az akciók megszerzése érdekében

* **Alkalmazott eszközök, módszerek:**

Az alkalmazás a Visual Studio Code nevű fejlesztői környezetben készült. A logika TypeScript nyelven íródott, a megjelenítés HTML és SCSS használatát követelte meg. A fejlesztésben egy multiplatformos keretrendszer, az Angular nyújtott segítséget, melynek a 13-as verzióját használtam. Adatbázisként, tárolóként, sőt autentikációt és hostingot biztosító eszköz gyanánt a Firebase felhőalapú szolgáltatás volt segítségemre. A projektet időnként a Git verziókezelő segítségével GitHubra feltöltve tárolom.

## Tartalomjegyzék

[*Feladatkiírás* 3](#_Toc138780823)

[*Tartalmi összefoglaló* 4](#_Toc138780824)

[Tartalomjegyzék 5](#_Toc138780825)

[1. Bevezetés 6](#_Toc138780826)

[1.1 Problémafelvetés, motiváció 7](#_Toc138780827)

[2. Alkalmazott eszközök, technológiák bemutatása 7](#_Toc138780828)

[2.1 Fejlesztői környezet 7](#_Toc138780829)

[2.1.1 Visual Studio Code 8](#_Toc138780830)

[2.1.2 Böngésző fejlesztői mód 8](#_Toc138780831)

[2.2 Programozási nyelv 9](#_Toc138780832)

[2.2.1 TypeScript 9](#_Toc138780833)

[2.2.2 HTML 10](#_Toc138780834)

[2.2.3 SCSS 10](#_Toc138780835)

[2.3 Keretrendszer 11](#_Toc138780836)

[2.3.1 Angular 11](#_Toc138780837)

[2.4 Firebase 11](#_Toc138780838)

[3. A megvalósítás lépései 12](#_Toc138780839)

# **Bevezetés**

A webáruházak egyre nagyobb népszerűségnek örvendenek. Az embereknek hatalmas kényelmi funkciót biztosít, hogy személyesen el sem szükséges menniük a termékekért, csupán csak rendelniük kell. Ennek okán az igény erre folyamatosan csak nő. Ezen webalkalmazás is többek között sablonként is szolgálhat újabban felépülő, hasonló célokra készített oldalak felépítésekor.

A cél az, hogy felhasználóbarát, minél gyorsabb, és minőségibb weboldalakat fejlesszünk, hisz az emberek türelmetlensége csökkentheti a vállalkozók szeme előtt lebegő bevételt, ha nem töltenek elég időt az adott weboldalon. A figyelemfelkeltés céljából ezen webáruház a szokványos hirdetések mellett egy beépített játékot fog tartalmazni, mellyel a felhasználó akciókra tehet szert. Ezzel elérhetjük azt, hogy több időt töltsön az oldalon, ezáltal több neki tetsző terméket tárhatunk a szeme elé, ami vásárlásra ösztönzi. Valójában ez egy nyerő-nyerő szituáció, hisz így elérjük, hogy a felhasználó vásároljon így profitot termelve a vállalkozónak. Még ha kevesebb is a profit az akciók miatt, a rendszer úgy lesz felépítve, hogy így is bőven nyereséges legyen.

A kész alkalmazás rendkívüli felhasználóbarátsága már nem igényel bármiféle szaktudást a kezeléshez. Egy laikus felhasználó is könnyen boldogulhat egyedül. Az admin felület kiépítése hasonló könnyedséget von majd maga után, ugyanis fontos, hogy az üzemeltető is gördülékenyen tudja működtetni az alkalmazást. A dinamikus megvalósítás lehetővé teszi, hogy könnyeden, mondhatni akármennyi ideig fusson alkalmazás, hisz a hosting-nak köszönhetően a világhálón bármikor megtekinthető. A hibák kezelése szintén beépített feladata az alkalmazásnak.

# **Problémafelvetés, motiváció**

A szakdolgozatom témája tehát az előző fejezetben felvázolt italárusító (mely természetesen más termékekkel is lehetne) webalkalmazás sablon elkészítése, egy egyszerű beépített Super Mario-hoz hasonló játékkal felvértezve. Korszerű technológia használatával, és folyamatosan bekerülő új funkciókkal.

A webáruház hozzájárulhat ahhoz, hogy szélesebb körben elterjedjen ez a fajta piaci rés kihasználás, miszerint egy játékkal csalogatjuk oda, sőt inkább tartjuk ott az oldalunkra tévedő felhasználókat.

A motivációm tehát annak a problémának a megoldására ad egy lehetőséget, miszerint a türelmetlen felhasználók túl hamar elhagynak egy adott webáruházat anélkül, hogy vásárolnának. Ez sokszor köszönhető a magas várakozási időnek míg betöltenek bizonyos részletek, a nem túl felhasználóbarát felületnek vagy épp a magas áraknak. Mindezek megoldása mellett még kedvezményt is biztosítok a vásárlók számára, amennyiben hajlandóak játszani. A játékok köztudottan addikciót okozhatnak, és ezen függőség kialakulását lehetővé téve akár napi szinten visszatérő vásárlóink is lehetnek.

# **Alkalmazott eszközök, technológiák bemutatása**

# **Fejlesztői környezetek**

A fejlesztői környezetek (IDE - Integrated Development Environment) olyan szoftvereszközök, melyek megkönnyítik a programozók munkáját. Olyasfajta segítséget nyújtanak, amik jelentősen lerövidíthetik egy-egy alkalmazás elkészítésének az idejét, így produktívabb és gyorsabb munkát eredményezve egyaránt.

Általában egy fejlesztő környezet több mindent magába foglal, többek között tartalmaz fordítót, futtatókörnyezetet, és a szoftver fejlesztéséhez elengedhetetlen szövegszerkesztőt egyben. Mondhatni egy integrált munkakörnyezet, melyben a programozók írhatnak, szerkeszthetnek, tesztelhetnek és hibát is kereshetnek a kódban, mindezt átlátható módon. Javaslatokat tesz, amikor elkezdünk gépelni egy adott parancsot, változónevet, függvénynevet, sőt képes a gépelési hibáink kijelzésére is.

A fejlesztői környezet sok mindent rejt magában, mint például szintaktikai kiemelés, kódszerkesztési segítség, hibakeresési eszközök, verziókezelési integráció és sok egyéb. Különböző programozási nyelveket és platformokat segítő variációi léteznek már, melyekből a programozó kiválaszthatja a saját igényeit és preferenciáit legjobban kielégítőt. A teljesség igénye nélkül néhány: Visual Studio Code, Eclipse, Notepad++, PyCharm, WebStorm stb.

# **Visual Studio Code**

Az általam használt Typescript nyelvhez a leginkább illeszkedő fejlesztői környezet a Visual Studio Code. Ez egy keresztplatformos eszköz, mely rengeteg programozási nyelvet és platformot támogat. Ezen felül sok egyéb fejlesztést segítő funkciót és bővítményt kínál. A Microsoft fejlesztette ki ezt a terméket, mely a kényelem megteremtését szolgálja, mint más fejlesztői környezetek, de tudomásom szerint a legszélesebb körű az alkalmazása. integráció és sok egyéb.

# **2.1.2 Böngésző fejlesztői mód**

Ha egy böngészőben aktiválni szeretnénk a fejlesztői módot csupán annyit kell tennünk, hogy az aktív ablakban lenyomjuk az F12 billentyűt, vagy jobb gombbal kattintunk az oldalon és kiválasztjuk a „Vizsgálat” menüpontot. Ekkor előjönnek különféle fejlesztést segítő eszközök (általában frontend fejlesztőknek nagy segítségére van). A legtöbb manapság használatos böngészőben már fellelhető ez a beépített funkció. Mindez lehetővé teszi, hogy az általunk éppen fejlesztett vagy mások akár mások által készített weboldalakat elemezzünk. Így akár könnyen látható, hogy hibakeresésre is alkalmas.

Több panellel rendelkezik, melyek különböző funkciókat tesznek lehetővé: elemek vizsgálata, a DOM szerkesztése, a CSS (stílusok), vagy akár a JavaScript módosítása, a hálózati lekérdezések monitorozását, a konzol üzeneteinek megtekintését és sok más eszközt, amelyek segítségével fejlesztés közben interaktívan dolgozhatunk a weboldallal.

# **Programozási nyelvek**

A programozási nyelvek afféle mesterséges nyelvek, melyeket azért hoztak létre, hogy az emberek utasítások megadásával tudjanak kommunikálni a számítógéppel. Értelemszerűen a gépek sokkal gyorsabbak, logikusabbak, mint az ember. Viszont közvetlenül nem képesek kommunikálni az emberekkel, hisz egy számítógép nem tud egyik emberi nyelven sem, mint ahogy nekünk sem egyszerű megérteni a számítógépek nyelvét. Szerencsére a programozási nyelvek megoldást nyújtanak ennek a problémának a megoldására.

A programozási nyelveknek saját szabályaik, szintaxisuk és funkcióik vannak. Ebben hasonlítanak az emberi nyelvekhez. Fontos, hogy egy programozási nyelv a programozni tanulók számára érthetővé válik, így biztosít könnyebb kommunikációt a számítógéppel. Általuk a fejlesztők írhatnak utasításokat, melyek sorozatából fel is épülnek a programokat és alkalmazásokat, amelyek képesek végrehajtani különféle feladatokat.

# **TypeScript**

A fejlesztés során kiemelt fontosságú, hogy milyen programozási nyelvet használunk, ugyanis előnyt jelenthetnek bizonyos programozási nyelvek az egyes feladatokat megvalósító alkalmazások elkészítésében. A TypeScript pedig rendkívül jól illeszkedik a Frontend fejlesztést segítő keretrendszerekhez, mint például az Angular, amit jelenleg használtam. A webalkalmazás logikája ezen a nyelven íródott így tehát több, mint fejlesztés felét kitette ennek a nyelvnek a használata. A TypeScript egy nyílt forráskódú, statikusan típusos objektumorientáltként is használható programozási nyelv, mely a JavaScript interpretált programozási nyelv hiányosságait terjeszti ki. Javarészt a Microsoft csapata fejlesztette, hogy megkönnyítsék a programozók munkáját. A TypeScriptet a fordító először JavaScript kóddá fordítja át, melyet már képes értelmezni a böngésző.

# **HTML**

A HTML (Hipertext Markup Language) egy leíró nyelv, ami a weboldalak struktúrájának felépítésére szolgál. Az oldal csupasz vázát felépíthetjük a HTML által előre definiált tag-párokkal, melyek egymásba ágyazásával egyedi struktúrát alakíthatunk ki. Értelemszerűen a Frontend fejlesztést segítő keretrendszereknek szüksége van valami ehhez hasonlóra. Többek között az Angular is a HTML használatát igényli azon felül, hogy valamennyi logikát is bevihetünk a használatával a HTML kódba.

# **SCSS**

Az SCSS (Sassy Cascading Style Sheet) egy stílusleíró nyelv, ami kiterjeszti a CSS-t, melybe előre definiált kulcs-érték párok megadásával tudjuk a HTML által felépülő struktúrának a megjelenését változtatni. Az SCSS könnyebb és hatékonyabb CSS stíluslapok írását teszi lehetővé, hisz kisebb logikai funkciókat biztosít, mint például a változók létrehozása. Egy preprocesszor dolgozza fel az SCSS fájlokat a fordítás részeként, és így hoz létre a háttérben CSS fájlokat.

# **Keretrendszerek**

A keretrendszerek arra szolgálnak, hogy a funkcióikat felhasználva lerövidítsék a fejlesztésre szánt időt. Nagyobb projektek esetén rendkívül hasznos, hisz ahogy a nevéből is adódik egy keretet ad az egész projektnek. Lényegileg olyan funkciókat tartalmaznak, melyeket alap esetben szintén le kéne fejleszteni, viszont a keretrendszereket adottak, és így bizonyos dolgok fejlesztését megspórolhatjuk. Ennek a haszna, hogy sokkal inkább koncentrálhatunk azon területek implementálására, mely az adott alkalmazásunk lényegét képezi.

# **Angular**

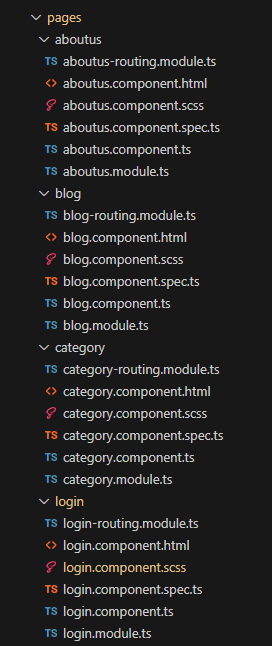
Az általam is használt nyílt forráskódú webes keretrendszer az Angular, melyet a Google fejlesztett ki, és adott ki 2010. október 20-án még Angular JS néven modern webalkalmazások fejlesztésének céljából. A célja, hogy felhasználóbarát, jól skálázható, hatékony alkalmazást fejlesszünk általa. Eleinte a JavaScript nyelvet használta a logikájához, de már az Angular 2-től áttért a JavaScriptet kiterjesztő TypeScript nyelv használatára. Biztosít adatkötést a felhasználói felület és a logika között, könnyű navigációt a Router segítségével és sablonok segítségével dinamikus és interaktív felhasználói felületek hozhatóak létre. Komponensekbe, avagy modulokba szervezhetjük az alkalmazásunkat a jobb átláthatóság végett. A segítségével nem csak webalkalmazások építhetőek fel, hanem például mobilalkalmazások, de még asztali alkalmazások is.

# **Firebase**

A Firebase a Google által kifejlesztett népszerű és teljeskörű fejlesztői platform. A gyors, hatékony fejlesztésben, tesztelésben, sőt az üzemeltetésben is nagy segítségére van a fejlesztőknek. Az eszközei és szolgáltatásai rengeteg segítséget nyújtanak. Idetartozik a felhasználókezelés és hitelesítés, a felhőalapú tárolás, a valós idejű adatbázis, de akár még az alkalmazásnaplózás is a teljesség igénye nélkül. A Firebase könnyen integrálható webes-, avagy mobilalkalmazásokba egyaránt. Az Angular keretrendszer is a TypeScript által könnyen képes alkalmazni a Firebase-t ezzel megkönnyítve a programozási munkálatokat.

# **A megvalósítás lépései**

# **3.1. Komponensek és modulok**

Egy Angular alkalmazás esetében elengedhetetlen, hogy komponensekbe szervezzük. Ez átláthatóbb kódot eredményez, és ezáltal külön egységekbe tudjuk tárolni az összetartozó részeket. A komponensek tehát külön egységeket jelölnek, melyek lehetővé teszik az újra felhasználhatóságot. Ezek adják a felhasználói felület (UI) építőköveit. Minden komponenshez tartozik logikáért felelős TypeScript fájl és a megjelenésért és stílusért felelős HTML és SCSS fájlok. A komponensek tehát felelősek felhasználói felület megjelenítéséért és a felhasználói interakciók kezeléséért. A komponenseket modulokba szervezhetjük, amik egy még nagyobb egységet alkotnak. Egy modul több komponensből is felépülhet, emellett szolgáltatásokat, csővezetékeket és egyéb fájlokat foglalhat magába. Az alkalmazás a legeneráláskor létre is hozza a gyökérmodult AppModule néven. Az ábrán látható néhány modul a saját alkalmazásomból.