Eötvös Loránd Tudományegyetem

Informatikai Kar

Algoritmusok és Alkalmazásaik Tanszék

Mintaillesztő algoritmusok szemléltetése

Szerző:

Ferencz-Sarmasági Csillag Ada

RZ6SNB

Programtervező informatikus BSc.

Témavezető:

dr. Ásványi Tibor

egyetemi docens, programtervező matematikus, informatikus PhD

Budapest, 2025

Tartalom

Bevezetés

Témabejelentő

Szakdolgozatomnak egy olyan szemléltető, tanító felület elkészítését választottam, amely a mintaillesztő algoritmusok működését, elméleti hátterét mutatja be egy weboldalon. Célja, hogy az algoritmusokkal ismerkedő diákok, például programtervező informatikus szakos hallgatók szemléletes módon konkrét példákon keresztül tanulmányozhassák a mintaillesztő algoritmusokat. A felhasználók saját mintaillesztéssel kapcsolatos feladataikat kipróbálhatják, ellenőrizhetik. A minta és a szöveg megadása után az oldal lépésről lépésre végig vezeti a felhasználót a kiválasztott algoritmus működése alapján. Fő feladatom a rendszer alapjainak lefektetése, az alkalmazás megjelenésének és felépítésének megtervezése és megvalósítása, melyhez a Google által fejlesztett Angular keretrendszert fogom használni. Ebben a rendszerben a felület felépítését HTML, az oldalakhoz tartozó design-t SCSS, a mögöttes logikát pedig TypeScript nyelven implementálom.

Témaválasztás indoklása

A témaválasztásomat mind személyes érdeklődésem, mind az egyetemi tanulmányaim során szerzett pozitív tapasztalataim motiválták. Az algoritmusok működésének megértése kiemelt szerepet kapott egyetemi pályafutásom során. Ezek az ismeretek segítettek kialakítani az algoritmikus gondolkodásomat. Az egyetemi képzés alatt több hasonló témájú projekt és szakdolgozat segítette elő tanulmányaimat, amelyek közvetlenül hozzájárultak hasonló algoritmusok működésének megértésében. Ezen felbuzdulva döntöttem úgy, hogy a saját szakdolgozatom is egy hasonló, interaktív tanulást elősegítő projekt legyen.

Nem csak azért választottam ezt a témát, mert az én tanulmányaimban fontos volt, hanem azért is, mert lehetőségem van arra, hogy visszaadjak valamit az egyetememnek. Szeretnék egy olyan eszközt létrehozni, amely hasonlóan, mint korábban nekem, más diákok számára is segítséget nyújt az algoritmusok megértésében. Egy jól strukturált, vizuálisan vonzó és interaktív weboldal ideális platformot biztosíthat a tanuláshoz. A diákok kísérletezhetnek, felfedezhetik az algoritmusok működését, és ezáltal jobban elsajátíthatják az algoritmusok alapelveit. Emellett a

felületen lehetőség nyílik egy kvízalapú gyakorlómodul segítségével is tesztelni a megszerzett tudást, így a hallgatók visszajelzést kaphatnak arról, mennyire értették meg a mintaillesztő algoritmusok működését.

Megoldandó feladat

A szakdolgozatom konkrét célja egy olyan weboldal létrehozása, amely három különböző mintaillesztő algoritmus – egyszerű mintaillesztés, QuickSearch és Knuth-Morris-Pratt (KMP) – működését mutatja be interaktív formában. Az oldal lehetővé teszi a felhasználók számára, hogy saját szöveget és mintáikat adják meg, majd egy szimuláció segítségével követhetik az algoritmusok folyamatát. Az algoritmusok minden egyes lépését részletesen szemléltetnek, magyarázatokkal kiegészítve, hogy a felhasználók pontosan érthessék, mi történik.

A weboldal funkciói között szerepelnek az interaktív gombok (indítás, leállítás, manuális irányítás, újrakezdés, algoritmus ismertetése), valamint a sötét és világos mód közötti választási lehetőség, amely segít a hosszú ideig tartó tanulmányozás során a szem kímélésében. A nyelvválasztási lehetőség (angol és magyar) pedig biztosítja, hogy az oldal nemzetközi diákok számára is hozzáférhető legyen.

A weboldal egy gyakorlófelülettel is kiegészül, amelyen keresztül a felhasználók kvízkérdések megválaszolásával ellenőrizhetik a megszerzett tudásukat. A kérdések az algoritmusok működésére, logikájára és gyakorlati alkalmazására fókuszálnak, így segítve a tanultak elmélyítését és az algoritmikus gondolkodás fejlesztését.

Ez a projekt tehát nem csupán egy technikai megvalósítás, hanem egy olyan oktatási eszköz, amely hozzájárul a diákok algoritmikus gondolkodásának és problémamegoldó képességeinek fejlesztéséhez. Az Angular keretrendszer használata lehetővé teszi, hogy az oldal reszponzív, gyors és könnyen kezelhető legyen, miközben a TypeScript nyelv biztosítja az alkalmazás stabilitását és biztonságát.

Felhasználói dokumentáció

Megoldott probléma

A szakdolgozatom konkrét célja egy olyan weboldal létrehozása, amely három különböző mintaillesztő algoritmus – egyszerű mintaillesztés, QuickSearch és Knuth-Morris-Pratt (KMP) – működését mutatja be interaktív formában. Az oldal lehetővé teszi a felhasználók számára, hogy saját szöveget és mintáikat adják meg, majd egy szimuláció segítségével követhetik az algoritmusok folyamatát. Az algoritmusok minden egyes lépését részletesen szemléltetnek, magyarázatokkal kiegészítve, hogy a felhasználók pontosan érthessék, mi történik.

A weboldal emellett egy beépített gyakorlómodult is tartalmaz, amely kvízkérdéseken keresztül segíti a tananyag feldolgozását és az ismeretek elmélyítését. A felhasználók önállóan ellenőrizhetik tudásukat az algoritmusokkal kapcsolatban, így aktívan reflektálhatnak a tanulási folyamatra.

Ez a megoldás különösen hasznos a programtervező informatikus szakos hallgatók számára, akik gyakran szembesülnek azzal a kihívással, hogy elméleti algoritmusokat kell megérteniük és alkalmazniuk kell a gyakorlatban. A szemléletes és interaktív bemutatás révén az oldal híd szerepet tölt be az elméleti tudás és a gyakorlati alkalmazás között. Ezzel segítve a hallgatókat az algoritmusok mélyebb megértésében és hatékonyabb használatában.

Az oldal tervezésekor kiemelt figyelmet fordítottam arra, hogy a felhasználói felület intuitív és felhasználóbarát legyen. A weboldal tartalmazza az alábbi főbb funkciókat:

- Interaktív elemek: Gombok, mint az indítás, leállítás, manuális irányítás, újrakezdés, és algoritmus ismertetése, amelyek lehetővé teszik a felhasználók számára, hogy aktív résztvevői legyenek a tanulási folyamatnak.
- Gyakorló oldal: Kérdések és válaszlehetőségek segítségével a felhasználók értékelhetik saját előrehaladásukat, és visszajelzést kapnak a megértés szintjéről.
- Két fajta design: Sötét és világos mód, amelyek segítik a hosszú távú koncentrációt és szem kímélését.
- Nyelvi opciók: Az oldal angol és magyar nyelven is elérhető, így nemzetközi és helyi diákok számára egyaránt hasznos.

A szakdolgozatom által megoldott probléma tehát egy sokoldalú oktatási platform létrehozása, amely segítségével a diákok nem csupán elméleti tudásra tehetnek szert, hanem ezt a tudást közvetlenül alkalmazni is tudják. Az interaktív szimuláció és a kvízmodul kombinációja lehetőséget ad a tanulás elmélyítésére és a tudás gyakorlati visszaigazolására. Ezáltal a projekt hozzájárul a diákok algoritmikus gondolkodásának és problémamegoldó képességeinek fejlesztéséhez, miközben modern webtechnológiák és programozási paradigmák alkalmazásával készül, mint az Angular, HTML, SCSS, és TypeScript, JSON. Az Angular keretrendszer használata biztosítja az alkalmazás stabilitását, gyorsaságát és skálázhatóságát. A TypeScript erős típusossága növeli a kód megbízhatóságát és karbantarthatóságát.

Weboldal elérése

A szakdolgozatom keretében kifejlesztett webalkalmazás eléréséhez az alábbi lépések szükségesek:

1. **Böngésző megnyitása:** Nyissa meg a preferált webböngészőjét, mint például a Google Chrome, vagy Safari.
2. **URL beírása:** Írja be a következő URL-címet a böngésző címsorába: mintailleszto-algoritmusok.hu
3. **Weboldal betöltése:** Nyomja meg az Enter billentyűt, és várja meg, amíg a weboldal teljesen betöltődik. A weboldal használatakor választhat a kívánt nyelv és megjelenési mód (sötét vagy világos) között.
4. **Navigáció az oldalon:** A weboldalon található navigációs menü segítségével választhatja ki, hogy melyik algoritmus működését szeretné vizsgálni. Az egyes algoritmusokhoz tartozó oldalakon interaktív funkciók segítik az algoritmusok megismerését és tesztelését, valamint lehetőség van rövid kvízek kitöltésére is a tanultak gyakorlásához.

Weboldal futtatása lokális környezetben

A projekt helyi gépen történő üzembe helyezése és futtatása a beadott fájlok alapján történik. Az alábbi lépések szükségesek a környezet előkészítéséhez és a projekt futtatásához:

1. **Fájlok letöltése:**
    - A projekt forráskódja és egyéb szükséges fájlok egy tömörített állományban lettek beadva. Töltse le ezt az állományt, és csomagolja ki egy kívánt helyre a számítógépén.

2. **Környezet előkészítése:**
    - **Node.js és npm telepítése:** A projekt Node.js környezetben fut, így szükséges a Node.js és a hozzá tartozó npm (node package manager) telepítése. A Node.js letölthető a Node.js hivatalos weboldaláról. A telepítés után ellenőrizze, hogy a Node.js és az npm megfelelően települt-e a következő parancsokkal:
      „node -v”
      „npm -v”
      Ezek a parancsok kiírják a telepített Node.js és npm verziókat.

3. **Navigálás a projekt könyvtárába:**
    - Miután kicsomagolta a projekt fájlokat, nyissa meg a terminált vagy parancssort, és navigáljon a projekt mappájába:
      „cd utvonal/projektmappa”

4. **Függőségek telepítése:**
    - A projekt könyvtárában futtassa az alábbi parancsot a szükséges npm csomagok telepítéséhez:
      „npm install”

      Ez a parancs telepíti az összes szükséges függőséget, amelyek a **package.json** fájlban vannak definiálva.

5. **Alkalmazás futtatása:**
    - Miután minden függőség sikeresen települt, indítsa el az alkalmazást a következő parancs segítségével:
      „npm start”

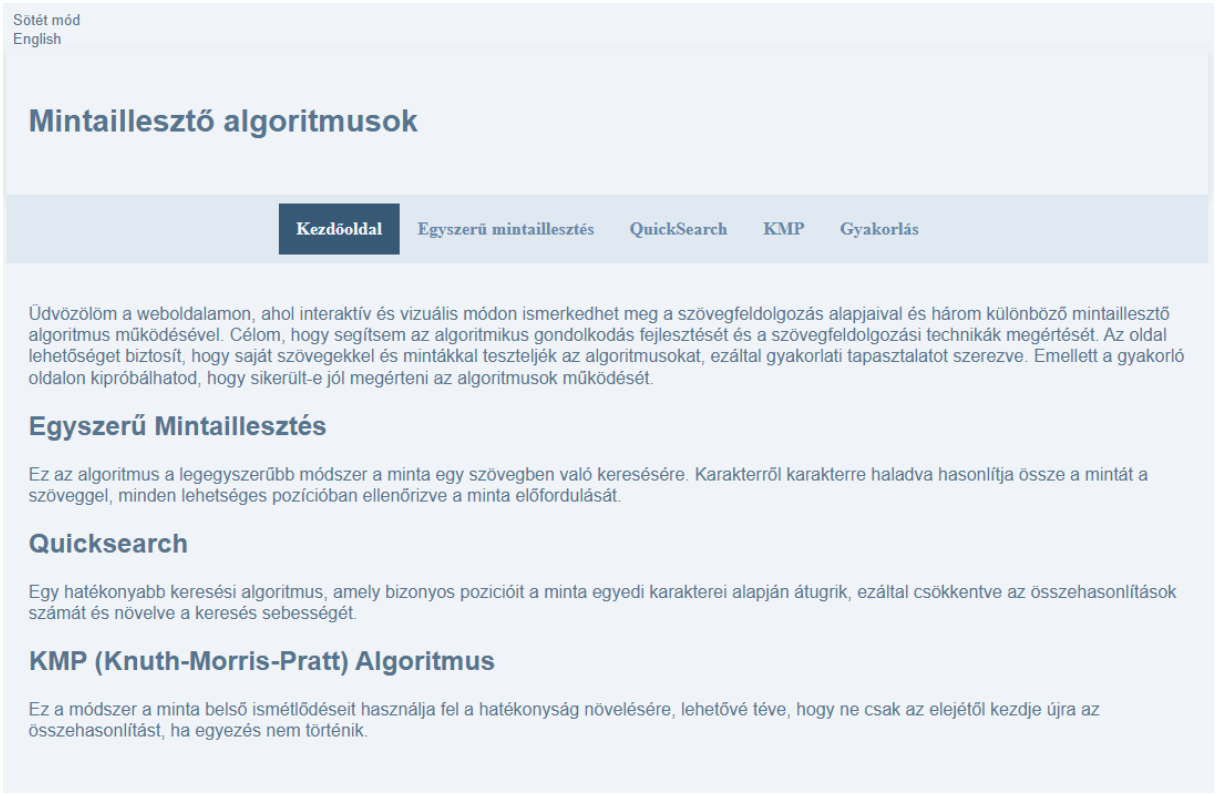
      Ez a parancs elindítja a fejlesztői szervert, és elérhetővé teszi az alkalmazást a **http://localhost:4200** címen a helyi gépen.

6. **Böngészőben való megtekintés:**
    - Nyissa meg a webböngészőt, és írja be a **http://localhost:4200** címet a címsorba, hogy elérje az alkalmazást.

Weboldal használata

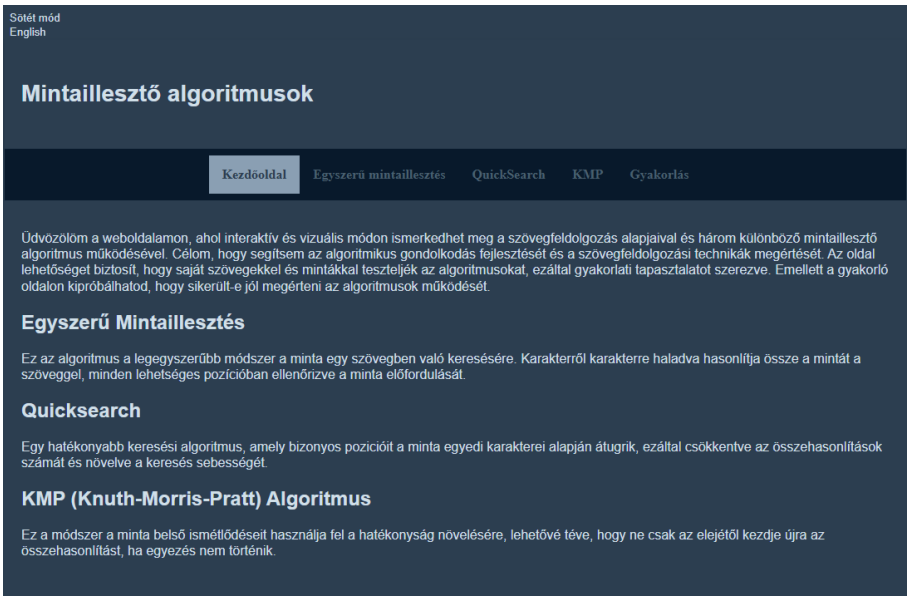
Kezdőoldal

A felhasználó a kezdőoldal elérését követően a következőt látja (1. ábra).



1.ábra kezdőoldal

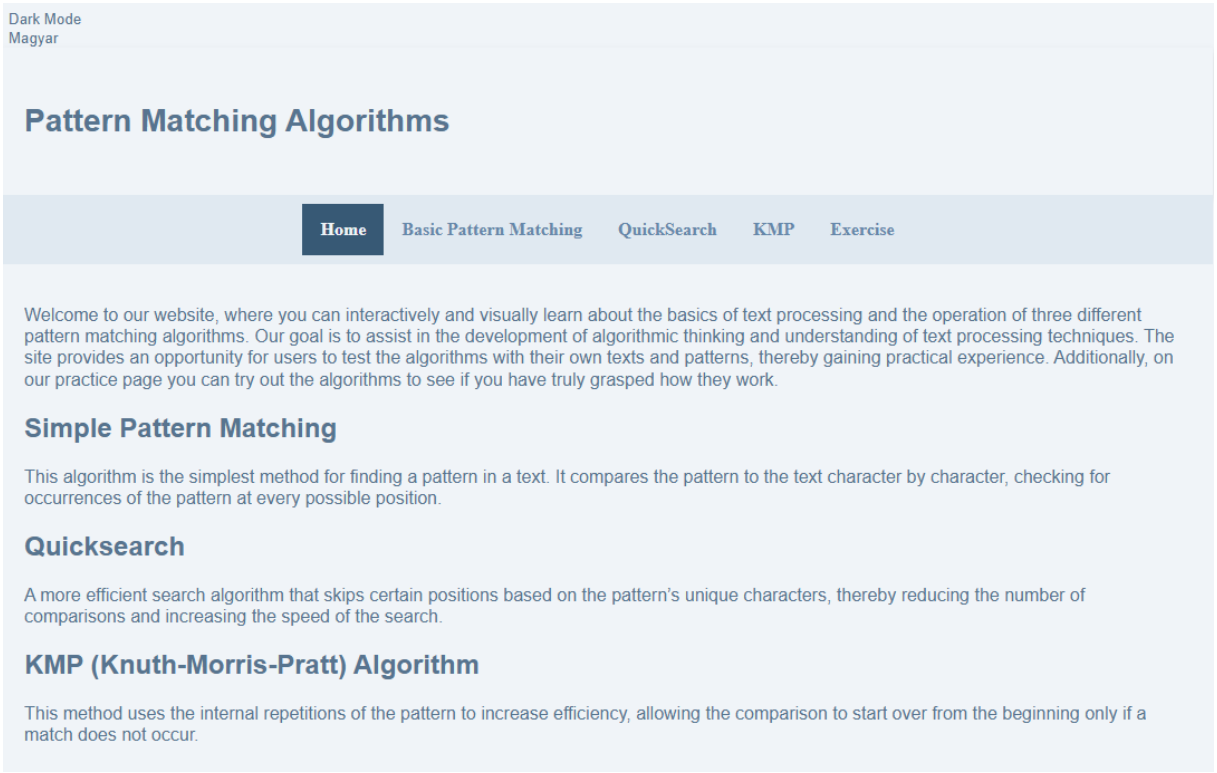
Sötét módra váltás: kattintson a weboldal bal felső sarkában lévő „Sötét mód” - feliratú gombra, és a weboldal azonnal átvált sötét témára. Ezt bármelyik oldalon megteheti, és bármikor vissza is lehet állítani, csak ismételten a „Sötét mód” -feliratú gombra kell kattintani. A sötét mód kiválasztását követően a következőt látja a felhasználó (2. ábra).



2. ábra. sötét mód

Átlépés egy kiválasztott oldalara: kattintson a menüben megjelenő oldalok valamelyikére, és a weboldal azonnal elnavigál a kiválasztott oldalra. Értelemszerűen ezt bármelyik oldalon megteheti akármikor, és mindig az az oldal fog megjelenni, amelyiknek a nevére kattintott.

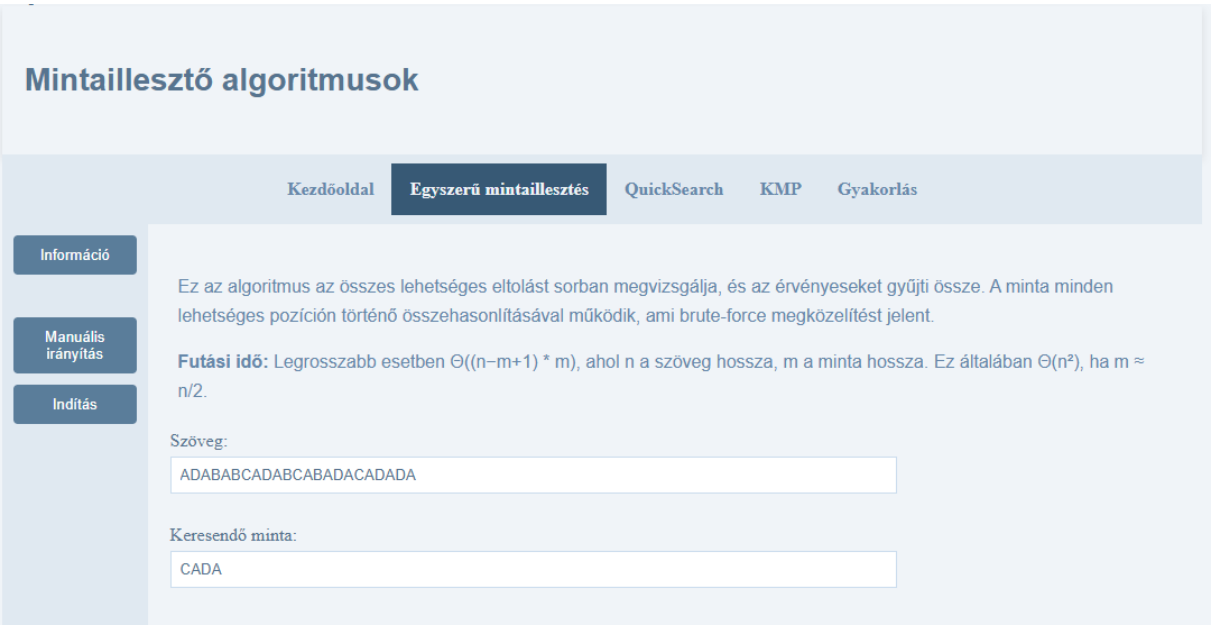
Angol nyelvre váltás: kattintson a weboldal jobb felső sarkában lévő „English” -feliratú gombra, és a weboldal azonnal átvált angol nyelvre (3. ábra). Visszaváltáshoz, kattintson újra ugyanarra a gombra („Magyar” -feliratú gombra).



3.ábra kezdőoldal angol módban

Közös funkciók minden algoritmus szimulációjánál

Információ megjelenítése: A bal oldalon található „Információ” gombra kattintva részletes leírás olvasható az adott algoritmus működéséről, futási idejéről. Újbóli kattintással az információs panel elrejthető (4. ábra).



4. ábra, információ megjelenítése egy mintaillesztést bemutató oldalon.

Saját szöveg, és minta megadása: a felhasználónak lehetősége van saját minta és szöveg megadására, ehhez a megfelelő input mezőre gépelje be, vagy illessze be a kívánt szöveget, és mintát. Ha nem ad meg a felhasználó egyéni szöveget, vagy mintát, akkor az alkalmazás az alapértelmezett szöveggel és mintával fog működni.

Manuális/automata mód közötti váltás: kattintson a „Manuális irányítás” -feliratú gombra, ha vissza szeretne váltani az automata módra kattintson az „Automata mód” -feliratú gombra. (5. ábra)

5. ábra automata mód gombok

Automata mód használata: indításhoz kattintson az „Indítás” -feliratú gombra, az indítást követően az algoritmus szemléltetésének a szimulációja elindul (6. ábra). Indítást követően lehetőség van a szimuláció leállítására a „Leállítás” -feliratú gomb megnyomásával. Leállítás után a felhasználó tudja folytatni („Folytatás”) a szimulációt (5. ábra). A felhasználó az automata mód közben bármikor dönthet úgy, hogy át akar váltani manuális irányításra, ehhez kattintson a „Manuális irányítás” – feliratú gombra. Ha a szimuláció véget ért a felhasználó tetszőlegesen újra indíthatja. Továbbá a szimuláció minden lepésnél magyarázó szöveget olvashat a felhasználó (6. ábra).

6. ábra. automata szimuláció

Manuális irányítás használata: első lépésben kattintson az „Előkészítés” - feliratú gombra. Majd tetszőlegesen tud haladni, vagy előre fele, a „Következő lépés” -feliratú gombbal, vagy (ha van megelőző lépés, akkor) hátrafele, az „Előző lépés” -feliratú gombbal. Az algoritmus bemutatásának lépéseit, a billentyűzetről is lehet irányítani (ha már elő lett készítve), a jobbra mutató nyíllal lehet a következő lépést elérni, a balra mutató nyíllal, meg az előző lépést. Ha a felhasználó manuális irányítás közben elölről szeretné kezdeni a lépéseket, kattintson ismét az „Előkészítés” -feliratú gombra (8. ábra).

7. ábra. vezérlő gombok

Egyszerű mintaillesztés oldal

Az egyszerű mintaillesztő algoritmus a szöveg minden egyes pozícióját megvizsgálja, és ellenőrzi, hogy a minta karakterei egyeznek-e a szöveg aktuális részletével. Az oldalon a folyamat vizuálisan követhető, amint egy találatot talál az algoritmus, az alkalmazás kiemeli a szöveg megfelelő részét és megjeleníti a találatot (8. ábra).

8. ábra. megtalált pozíció

Ez a módszer ugyan egyszerű és megbízható, azonban nem hatékony, mivel minden pozícióban el kell végezni az összehasonlítást, így időigényes lehet nagyobb szövegek esetén. A műveletigénye a legrosszabb esetben $\Theta((n-m+1)\cdot m)$, amikor minden karaktert részletesen össze kell hasonlítani, és a legjobb esetben $\Theta(n-m+1)$, ha a minta már az első karaktereknél kizárható.

Az egyszerű mintaillesztési technikát számos területen alkalmazzák: plágiumkeresés során dokumentumok összevetésére, spam szűrésben kulcsszavak keresésére, szövegszerkesztők helyesírás-ellenőrző funkciójában, valamint bioinformatikában DNS szekvenciák azonosítására.

QuickSearch oldal

A QuickSearch algoritmus első lépésként kiszámolja a minta karaktereihez tartozó shift értékeket, amelyek meghatározzák, hogy egy sikertelen összehasonlítás után mennyivel kell eltolni a mintát. A weboldalon a shift-tábla automatikusan generálódik a minta alapján, és megjelenik a szimuláció előtt. A shift értékek kiszámításának magyarázatához a felhasználó az egérmutatót a táblázat fölé helyezheti, ilyenkor szöveges magyarázat jelenik meg, amely lépésről lépésre elmagyarázza, hogyan számolódtak ki az értékek (9. ábra).

9. ábra: shift tábla magyarázat

A QuickSearch algoritmus hatékonyságát az adja, hogy az egyszerű mintaillesztéssel szemben nem minden pozícióban végzi el az összehasonlítást. Amint eltérés van, a kiszámolt shift-érték alapján lép előre a szövegben. A teljes műveletigénye így általában sokkal kedvezőbb, mint az egyszerű mintaillesztésé, és O(n) időben működik átlagos esetekben.

A QuickSearch algoritmushoz hasonló hatékony mintakeresési eljárásokat használnak például szövegindexelők, keresőmotorok, adatbázisok gyors szövegkeresési műveleteiben, illetve helyesírás-ellenőrzők és víruskereső rendszerek szöveges szűrési feladataiban.

KMP oldal

A Knuth-Morris-Pratt (KMP) algoritmus első lépésként egy úgynevezett π (pi) táblázatot számol ki, amely megmutatja, hogy egy részleges egyezés esetén hány karaktert lehet átugrani a mintában. A weboldalon a π-tábla kiszámítása lépésről lépésre követhető, a szimuláció minden lépésnél megmagyarázza, hogy az adott érték hogyan jött ki (10. ábra).

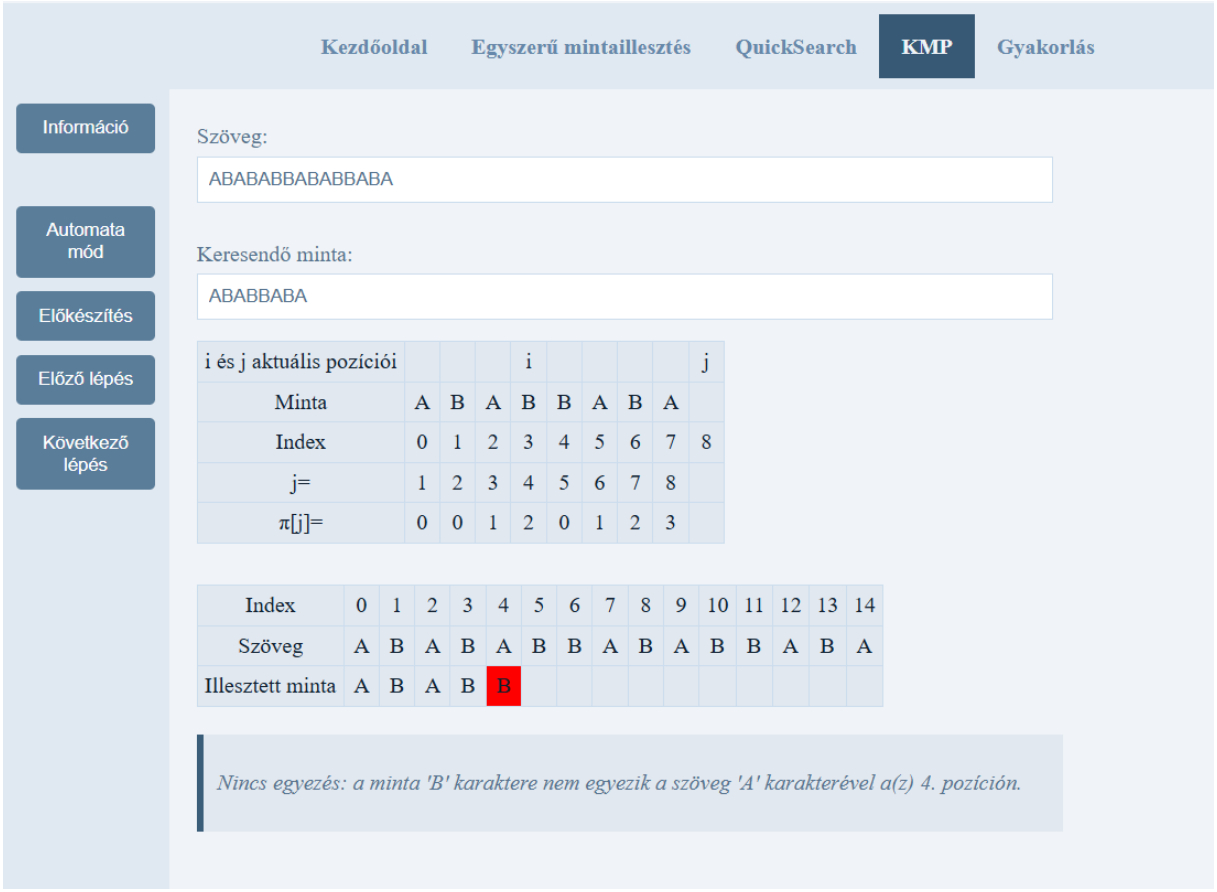
10. ábra: pi kalkuláció

Amennyiben a felhasználó kíváncsi a π-tábla kiszámításának stuktogramjára[1], húzza az egérmutatót a táblázat fölé és megjelenik (11. ábra).

11. ábra: struktogram megjelenítése

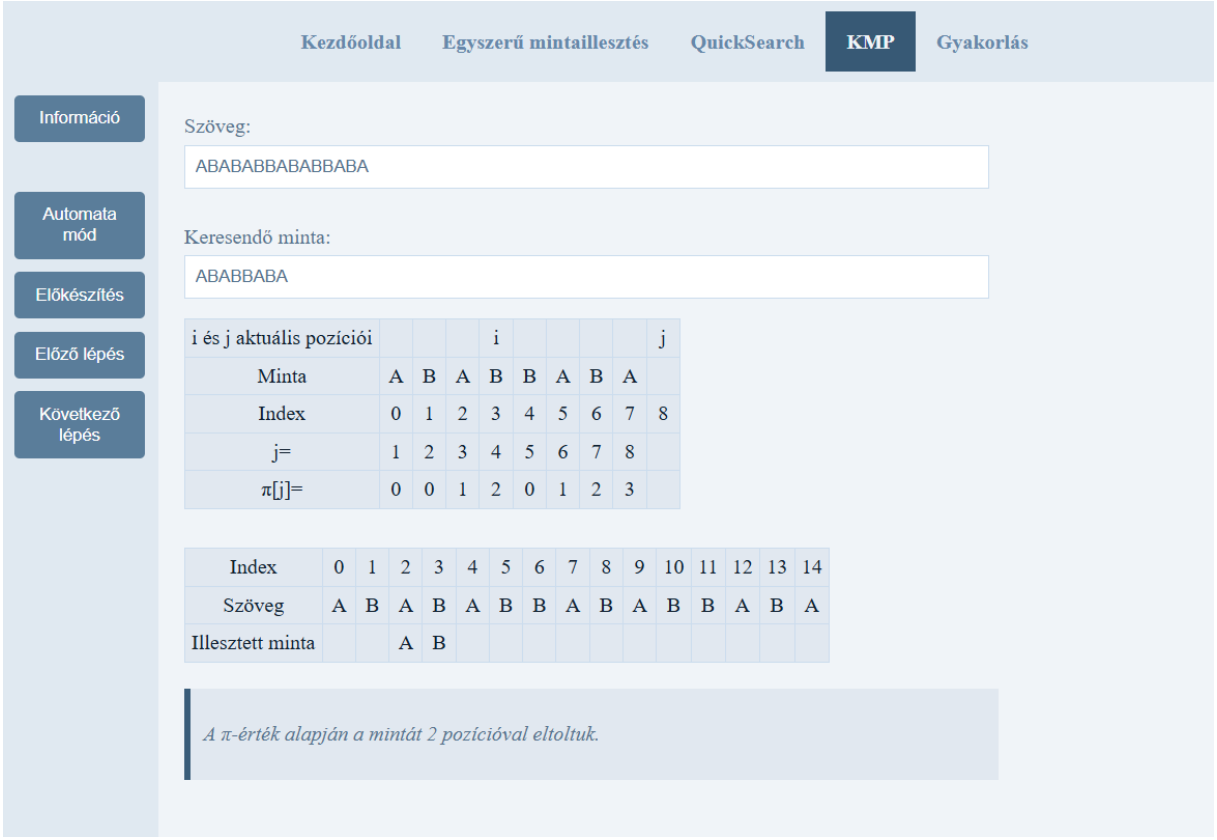
1 dr. Ásványi Tibor: Algoritmusok és adatszerkezetek II. előadásjegyzet

Miután a π-tábla elkészült, megkezdődik a keresési fázis. A szimuláció során a karaktereket egyesével helyezi le az algoritmus. Ha egy karakter nem egyezik, az algoritmus a korábban kiszámolt π értékek alapján hajtja végre az eltolást. Ezzel jelentősen csökkentve a felesleges összehasonlításokat (12. ábra, 13. ábra).



12. ábra: nem egyező karakter

Továbbá, a keresési folyamatot is vizuálisan kiegészíti egy struktogram[2], ha a felhasználó az egérmutatót a mintaillesztést végző táblázat fölé helyezi, akkor megjelenik a mintaillesztési folyamat struktogramja is, amely bemutatja a keresés logikai lépéseit.



13. ábra: eltolás

A KMP algoritmus műveletigénye Θ(n), mivel a minta előfeldolgozása és a keresési fázis is lineáris időben zajlik. Ez jóval hatékonyabbá teszi az algoritmust, különösen hosszú szövegek és minták esetén.

A KMP algoritmust széles körben alkalmazzák például szövegkereső rendszerekben, dokumentumindexelésnél, adatbázis-kezelők szöveges lekérdezéseinél, valamint bioinformatikai elemzések során DNS és fehérjeláncok szakaszainak gyors keresésére.

2 dr. Ásványi Tibor: Algoritmusok és adatszerkezetek II. előadásjegyzet

Gyakorlás oldal

A felhasználó a Gyakorlás oldal elérése után a következőt látja. (14. ábra)

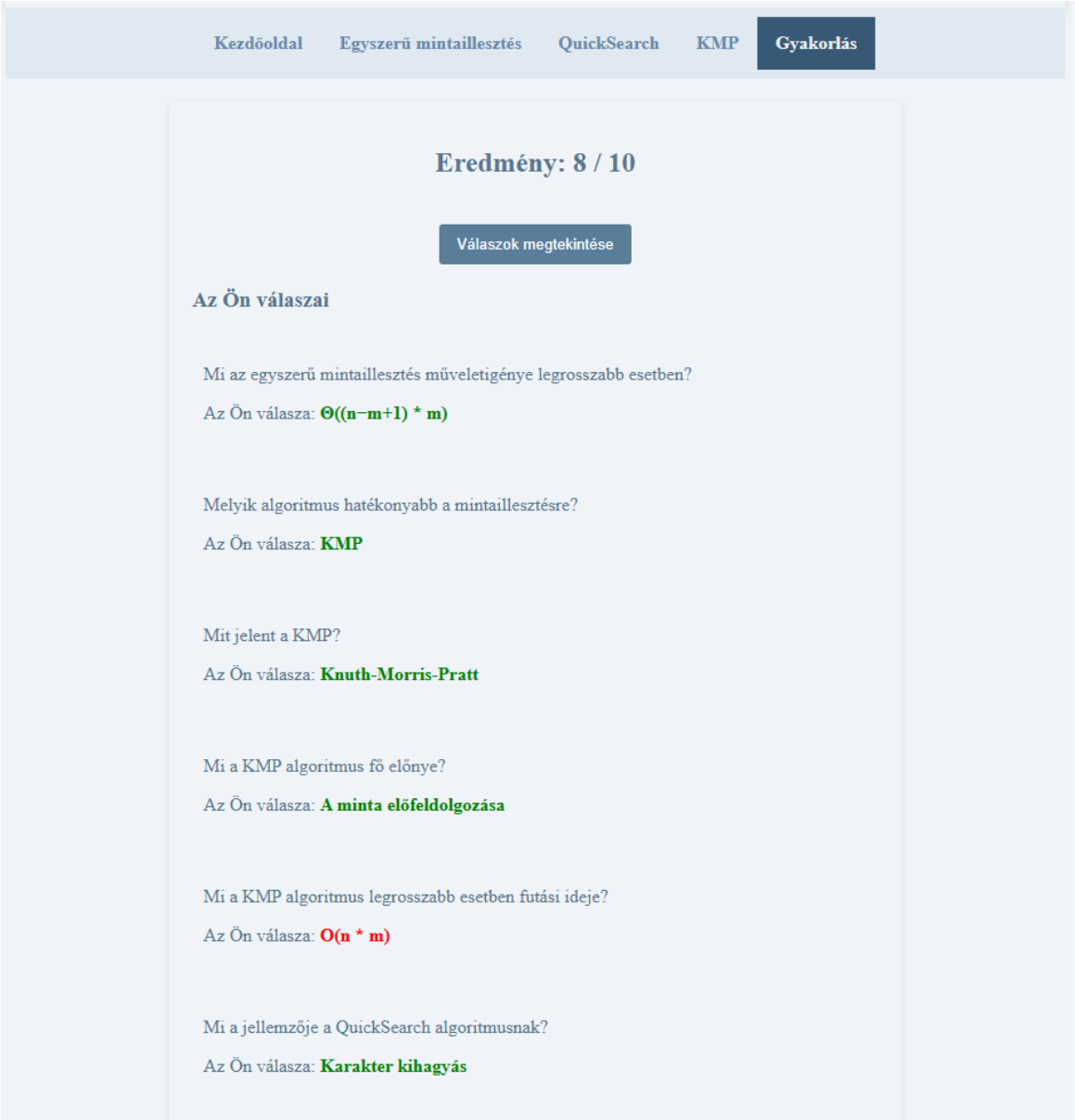
1. **Kvíz elindítása:** Kattintson a felső navigációs sávban a „Gyakorlás” menüpontra. A megnyíló oldalon kérdések jelennek meg, mindegyikhez több válaszlehetőséggel. A felhasználó válassza ki azt, amelyiket helyesnek gondolja. (14. ábra)

14. ábra Gyakorló oldal

2. **Válaszadás és automatikus továbblépés:** Amint a felhasználó rákattint egy válaszlehetőségre, az alkalmazás automatikusan a következő kérdésre ugrik.
3. **Eredmények:** Az utolsó kérdés megválaszolása után automatikusan megjelenik a kitöltött kvíz eredménye. (ábra 15.)

15. ábra: kvíz eredménye

4. **Válaszok kiértékelése**: A kvíz befejezése után megjelenik a „Válaszok megtekintése” – feliratú gomb, amennyiben kíváncsi a válaszai helyességére, kattintson erre a gombra. A felületen színek jelölik a válaszok helyességét, a helyes válaszokat zöld, a helytelenül megjelölteket piros színnel emeli ki a rendszer. A felhasználó minden kérdésnél láthatja, mit választott. (16. ábra)



16. ábra: megadott válaszok kiértékelése

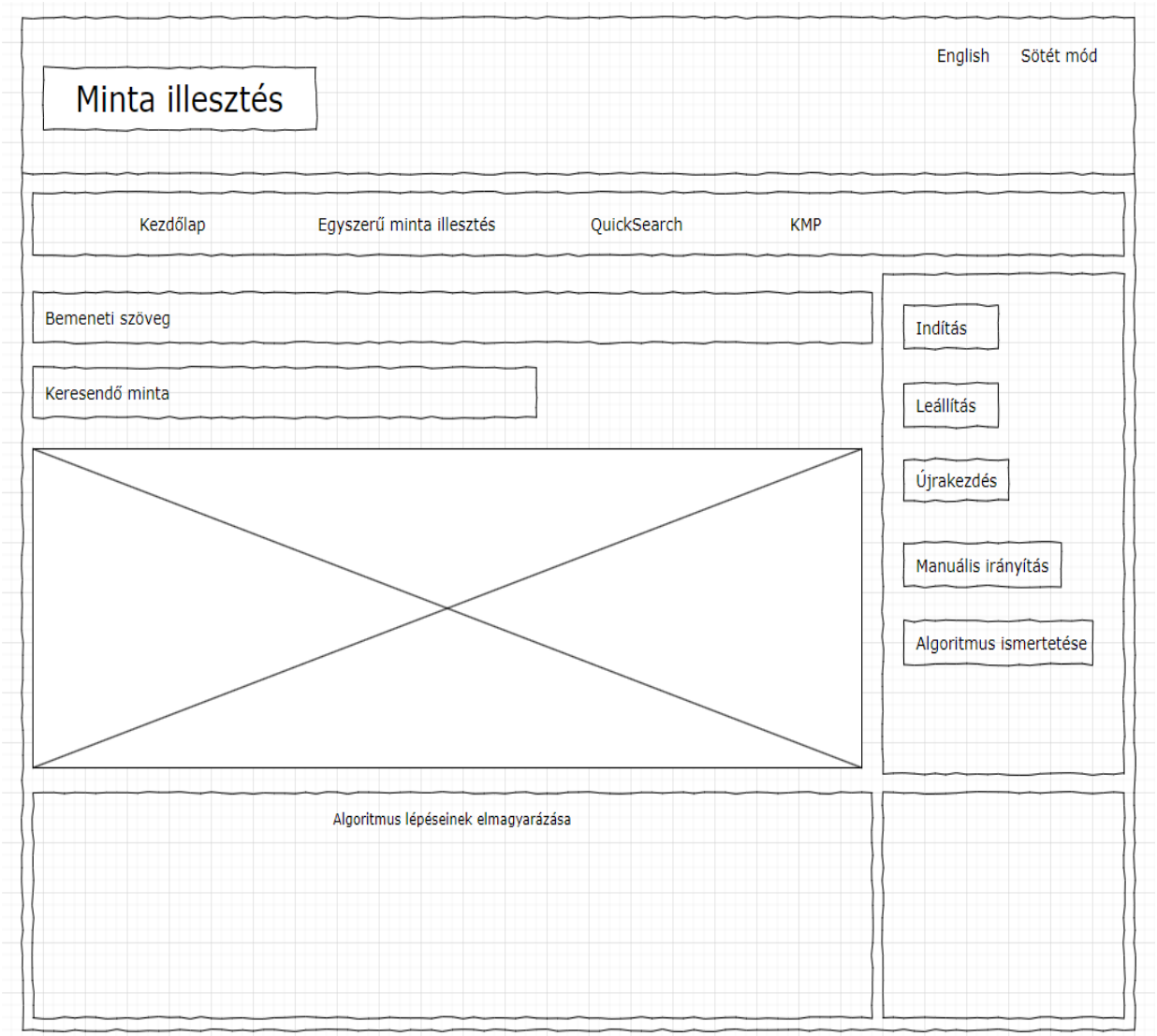
Fejlesztői dokumentáció

Tervek

Felület terv

A fejlesztés kezdeti szakaszában elkészítettem egy vázlatos felület tervet, amely a weboldal felépítésének alapjait határozta meg. A célom az volt, hogy áttekinthető, könnyen használható, és az algoritmusok működését jól szemléltető struktúrát alakítsak ki.

Az alábbi képen látható a kezdeti felületterv (17. ábra).

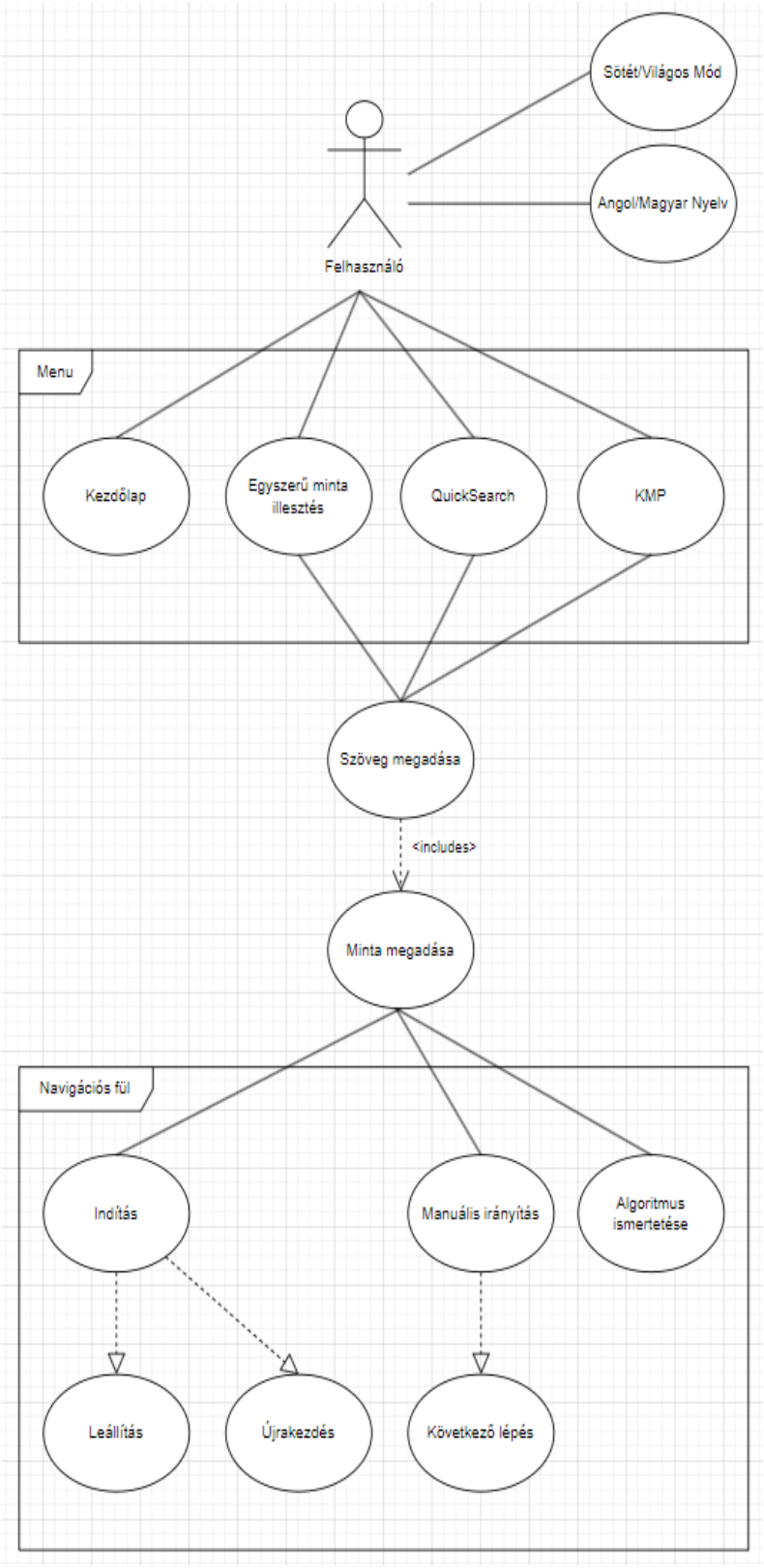


17. ábra: felület terv

A fejlesztés során ezt a tervet alapul véve készítettem el a végleges felületet, amely néhány ponton továbbfejlődött, és a gyakorlati szempontokhoz igazodott.

Use-case diagram

A rendszer funkcionális követelményeinek feltérképezése érdekében egy use-case diagramot készítettem, amely áttekinthető formában szemlélteti a felhasználó és a rendszer közötti lehetséges interakciókat. A diagram segítségével jól láthatóvá válik, hogy a felhasználó milyen főbb funkciókhoz férhet hozzá, valamint hogyan épülnek egymásra az egyes műveletek.



18. use-case diagram

A következő ábra a rendszer kezdeti funkcionalitásának modelljét mutatja be (18. ábra).

A tervezés során kiemelt figyelmet fordítottam arra, hogy a rendszer kezelése logikus felépítésű, átlátható és könnyen követhető legyen a felhasználók számára. A későbbi fejlesztések során a use-case diagram alapvető struktúráját megtartottam, ugyanakkor a gyakorlati tapasztalatok alapján kisebb finomításokat végeztem rajta: például a navigációs lehetőségeket bővítettem, és az irányítást még intuitívabbá tettem, hogy a felhasználói élményt tovább javítsam.

Követelmény leírás

Ezen weboldal fő célja, hogy interaktív és vizuális formában mutassa be három különböző mintaillesztő algoritmus működését, azáltal, hogy lehetőséget biztosít a felhasználóknak saját szöveg és minta bevitelére. Az oldal kifejezetten diáktársaim számára készül, hogy könnyebbé tegye számukra az algoritmikus gondolkodás és a szövegfeldolgozás elveinek megértését.

Kezdőlap:

-Rövid áttekintés a három mintaillesztő algoritmusról.

-Sötét/Világos mód közötti választás

-Angol/Magyar nyelv közötti választás

Egyszerű Mintaillesztés Oldal:

-Lehetőség a felhasználóknak saját szöveg és minta megadására.

-Interaktív gombok: Indítás, Leállítás, Manuális Irányítás, Újrakezdés, Algoritmus Ismertetése

-A szimuláció során a megadott szöveg alatt mozgatjuk a keresett mintát, úgy, hogy a lehető legjobban megértesse a felhasználóval az algoritmus működését.

-A szimuláció alatt, minden lépésnek a szöveges elmagyarázása is megjelenik.

-Sötét/Világos mód közötti választás

-Angol/Magyar nyelv közötti választás

QuickSearch Oldal:

-Lehetőség a felhasználóknak saját szöveg és minta megadására.

-Interaktív gombok: Indítás, Leállítás, Manuális Irányítás, Újrakezdés, Algoritmus Ismertetése

-A szimuláció előtt kirajzolja a shift-értékek táblázatát, majd a szimuláció során a megadott szöveg alatt mozgatjuk a keresett mintát, úgy, hogy a lehető legjobban megértesse a felhasználóval az algoritmus működését.

-Lehetőség a shift-tábla működésének megértésére

-A szimuláció alatt, minden lépésnek a szöveges elmagyarázása is megjelenik.

-Sötét/Világos mód közötti választás

-Angol/Magyar nyelv közötti választás

KMP Oldal:

-Lehetőség a felhasználóknak saját szöveg és minta megadására.

-Interaktív gombok: Indítás, Leállítás, Manuális Irányítás, Újrakezdés, Algoritmus Ismertetése

-A szimuláció előtt kiszámolja a π függvényt, majd a szimuláció során a megadott szöveg alatt mozgatjuk a keresett mintát, úgy, hogy a lehető legjobban megértesse a felhasználóval az algoritmus működését.

-A szimuláció alatt, minden lépésnek a szöveges elmagyarázása is megjelenik.

-Sötét/Világos mód közötti választás

-Angol/Magyar nyelv közötti választás

Gyakorló Oldal:

-Válaszlehetőségek közül való választás egy-egy kérdésre.

-Minden válasz után a rendszer automatikusan továbblép a következő kérdésre.

-Az utolsó kérdés megválaszolása után megjelenik a „Válaszok megtekintése” gomb.

-A válaszok megtekintése oldalon a felhasználó láthatja, melyik választ jelölte meg, és melyik volt helyes.

-A helyes válaszok zöld, a helytelen válaszok piros színnel jelennek meg.

Felhasználói előnyök:

-Vizuális tanulás: A szimulációk és animációk segítségével a felhasználók könnyen megérthetik az algoritmusok működését.

-Interaktivitás: A gombok segítségével a felhasználók maguk irányíthatják és tanulmányozhatják az algoritmusokat.

-Gyakorlati tapasztalat: A saját szöveg és minta beviteli lehetőség révén a felhasználók valós időben gyakorolhatják az algoritmusok alkalmazását.

- Önellenőrzés és önfejlesztés lehetősége: A beépített kvízkérdések segítségével a felhasználók visszajelzést kaphatnak a tudásukról, ezáltal mélyebb megértést alakíthatnak ki, és aktívan bevonódhatnak a tanulási folyamatba.

-Sötét/világos mód a szem kímélése érdekében

-Angol nyelvre váltás külföldi hallgatók számára

Ezen a weboldalon keresztül az algoritmusok iránt érdeklődő diákok egy intuitív és interaktív módon mélyülhetnek el a mintaillesztés világában, fokozva az algoritmikus gondolkodásukat és megértésüket.

Felhasznált módszerek

A szakdolgozatomban bemutatott webalkalmazás fejlesztése során számos modern szoftverfejlesztési módszert és technológiát alkalmaztam annak érdekében, hogy a projekt céljait hatékonyan és professzionálisan valósíthassam meg. Az alábbiakban részletezem a projekt megvalósításához felhasznált főbb módszereket és technológiákat.

1. Agilis fejlesztési módszertan

Az agilis szoftverfejlesztési keretrendszerek, mint a Scrum és Kanban, lehetővé teszik a rugalmas tervezést, gyors prototípuskészítést és az iteratív fejlesztést. A sprintekre bontott munka segítségével hatékonyan tudtam kezelni a fejlesztési feladatokat és prioritásokat, miközben biztosítottam a projekt időben történő és sikeres befejezését.

2. Webes technológiák

A webalkalmazás a következő technológiákat használja:

- **HTML5**: Az oldal szerkezetének felépítésére használom, modern, szemantikus elemekkel, amelyek támogatják az oldal akadálymentesítését és SEO-optimalizálását.
- **SCSS (Sassy CSS)**[3]: Lehetővé teszi változók, keverékek (mixins), és funkciók használatát, így a stíluslapok karbantarthatóbbak és átláthatóbbak maradnak.
- **TypeScript**[4]: A JavaScript szigorúan típusos verziója, amely növeli a fejlesztési folyamat során a kód megbízhatóságát és az alkalmazás hibatűrését.
- **JSON (JavaScript Object Notation):** Könnyűsúlyú, szöveges adatcsere-formátum, amely jól strukturált, ember által is olvasható módon tárolja az adatokat. Egyszerűen integrálható különféle alkalmazásrészek közé, és támogatja a dinamikus tartalomkezelést.

3 Sass Dokumentáció: https://sass-lang.com/documentation
4 TypeScript Dokumentáció: https://www.typescriptlang.org/docs/

3. Angular keretrendszer[5]

Az Angular egy komponens-alapú keretrendszer, amelyet a Google fejlesztett. Nagy mértékben elősegíti a modern, interaktív webes alkalmazások fejlesztését. Az Angular következő jellemzőit használtam ki a projekt során:

- **Komponens-alapú architektúra**: Lehetővé teszi az alkalmazás logikájának és felületének moduláris felépítését, amely elősegíti a kód újra felhasználását és tesztelhetőségét.
- **Szolgáltatások és injektálás (Services and Dependency Injection)**: Lehetővé teszi a függőségek egyszerű kezelését és a kód moduláris szerkezetének fenntartását.

4. Reszponzív design

A Bootstrap és más reszponzív keretrendszerek segítségével biztosítottam, hogy az alkalmazás különböző készülékeken (asztali számítógépek, táblagépek, okostelefonok) is jól használható legyen. A média lekérdezések és rugalmas elrendezések biztosítják, hogy a felhasználói felület minden eszközön optimális legyen.

5. Algoritmusok implementálása

Az algoritmusok implementálásához a TypeScript által nyújtott előnyöket használtam ki. Az egyszerű mintaillesztés, QuickSearch és KMP algoritmusokat tisztán és hatékonyan implementáltam, figyelemmel a futási időre és memóriahasználatra.

A projekt során alkalmazott technológiák és módszertanok mellett a szakdolgozat fejlesztéséhez további eszközöket és platformokat is igénybe vettem a hatékony együttműködés, verziókezelés és üzemeltetés érdekében. Ezek a következők:

6. GitLab[6]

A GitLab-ot használtam a projekt verziókezelésére és a forráskód tárolására. A GitLab egy nyílt forráskódú verziókezelő rendszer és CI/CD platform, amely lehetővé teszi a fejlesztési folyamatok automatizálását. Továbbá a kód változásainak nyomon követését és a feladatok kezelését. A GitLab integrált funkcionalitásaival, mint a hibakövetés, kódellenőrzés és automatizált tesztelés, biztosítottam, hogy a fejlesztési lehetőségek során a kód minősége folyamatosan magas szinten maradjon.

---

5 Angular Dokumentáció: https://angular.io/docs
6 GitLab Dokumentáció: https://docs.gitlab.com

7. Amazon Web Services (AWS)[7]

Az alkalmazást az Amazon Web Services (AWS) felhőalapú infrastruktúráján üzemeltettem. Ez biztosítja a skálázhatóságot, a rendelkezésre állást és a biztonságot. Az AWS szolgáltatások közül elsősorban az Amazon EC2 (Elastic Compute Cloud) példányokat használtam az alkalmazás tárolására és futtatására

Ezek a technológiák és eszközök együttesen biztosították, hogy a projekt nemcsak technikailag magas színvonalon valósuljon meg, hanem a fejlesztési és üzemeltetési folyamatok is gördülékenyen, hatékonyan és biztonságosan zajljanak. A GitLab és az AWS felhasználása nélkülözhetetlen volt a projekt sikeréhez. Lehetővé tették a kód szervezett kezelését, a folyamatos integrációt és a magas rendelkezésre állású üzemeltetést.

Szoftver szerkezete

Fájlstruktúra

A projekt mappastruktúrája világosan elkülöníti a statikus erőforrásokat, a fordítási és build kimeneteket, valamint az alkalmazás logikai felépítését támogató komponenseket. Az alábbiakban ismertetem a legfontosabb könyvtárakat:

assets: i18n: Tartalmazza a nemzetközi fordítási forrásfájlokat (például en.json, hu.json). Ezek a JSON formátumú fájlok kulcs-érték párokat tartalmaznak, amelyek segítségével az alkalmazás több nyelven is elérhetővé válik.

coverage: Ebben a könyvtárban kerülnek elmentésre a tesztlefedettségi jelentések. Ezek segítik a fejlesztőt abban, hogy átlássák, mely részek lettek tesztelve, és hol szükséges további fejlesztési beavatkozás.

dist: Az Angular build folyamatának eredményeit tartalmazza. Itt található a teljesen összefűzött, optimalizált és produkcióra szánt alkalmazás, mely a végfelhasználó által futtatható.

[7] *Amazon Web Services Dokumentáció: https://aws.amazon.com/documentation/*

node_modules: Az alkalmazás számára szükséges külső függőségek, könyvtárak telepített példányai találhatók itt. Az npm által kezelt csomagok elérhetősége ebben a könyvtárban biztosított.

pages

A projekt fő felhasználói felületeit megvalósító komponenseket tartalmazza, amelyeket az alábbiak szerint csoportosítottunk:

Home oldal: Az alkalmazás kezdőoldala, amely bemutatja az általános célokat, és navigációs lehetőségeket biztosít az egyes funkcionális modulok eléréséhez.

Három mintaillesztő oldal: Mindegyik oldal egy adott mintaillesztő algoritmus megvalósítását demonstrálja. Ezek az oldalak tartalmazzák az algoritmus lépésenkénti vizuális bemutatását, illetve a felhasználó által bevitt szöveg- és mintaadatok alapján történő interakciót.

Gyakorló oldal: Ez az oldal lehetőséget ad a felhasználónak az algoritmusok interaktív kipróbálására. A beviteli mezők és a vizuális visszajelző elemek segítségével tesztelhető a mintaillesztés folyamata.

Komponens Architektúra

Az Angular komponens alapú felépítése lehetővé teszi, hogy az alkalmazás logikailag elkülönített, újra felhasználható egységekből épüljön fel. Az alábbiakban láthatóak a fő komponensek és azok rövid ismertetése:

HomeComponent

Funkció:

- Az alkalmazás bevezető oldala, weboldal ismertetése, átfogó leírás a mintaillesztő algoritmusokról.

- Navigációs lehetőség a különböző oldalak között. (mindegyik oldal tartalmazza)

ExcersiseComponent (Gyakorló oldal)

Funkció:

- Lehetővé teszi a felhasználók számára, hogy egy kvíz megoldásával teszteljék sikerült-e elsajátítaniuk a mintaillesztő algoritmusok működését.

- Visszajelzést kap a felhasználó a kvíz eredményéről, megjelölve a helyes és helytelen válaszokat.

PatternMatchingComponent-ek (Három mintaillesztő oldal)

Funkció:

- Mindegyik komponens egy specifikus mintaillesztő algoritmus implementációját tartalmazza (egyszerű mintaillesztés, QuickSearch, KMP).

- A komponensek felelősek az adott algoritmus logikájának végrehajtásáért, beleértve az alábbi metódusokat:

    - setDefaultText és setDefaultPattern: Alapértelmezett szöveg és minta beállítása a bemutatóhoz.

    - prepareMatching: Előkészíti a komponens állapotát egy új futáshoz. Törli az előző eredményeket, inicializálja a megjelenített tömböt (displayedText), és meghívja az algoritmus-specifikus előfeldolgozást (pl. shift-tábla, π-tömb).

    - highlightNextMatch: Az algoritmus lépésenkénti végrehajtását segítő metódus, amely a találatok vizuális kiemeléséért felel.

    - restart: A komponens állapotának alaphelyzetbe állítása, a szimuláció újbóli elindítása.

    - Segédellenőrző függvények: isFullMatchAtIndex, isMatch, getPatternCharAt, getCellColor, continueMatching

NavigationComponent

Funkció:

- Az szimulációk irányításáért felelős gombokat tartalmazza, amelyeken keresztül a felhasználó az interaktív bemutató során kedve szerint irányíthatja a szimulációt.

Moduláris Felépítés és Interakciók

Angular Modulok

Routing Modul: Az alkalmazás oldalai közötti navigációját a RouterModule kezeli, amely konfigurációja biztosítja, hogy a megfelelő komponens jelenjen meg a felhasználói navigáció hatására.

FormsModule: A felhasználói űrlapok kezeléséért felelős modul, amely az adatbindingot és az űrlap vezérlést valósítja meg. Ez alapvető a felhasználó által bevitt szöveg- és mintaadatok feldolgozásában.

TranslateModule: Támogatja a többnyelvűséget az alkalmazásban az i18n mappa tartalmának betöltésével, lehetővé téve a dinamikus nyelvi váltást.

Interakciós Mechanizmusok

Menü:A MenuComponent vezérli az oldalak közötti váltást, ahol a menüstruktúra biztosítja a gyors hozzáférést a Kezdőoldal, Egyszerű mintaillesztés, QuickSearch, KMP, Gyakorlás oldalak között.

Adatfeldolgozás és Visszajelzés: A mintaillesztő komponensek belső logikája felelős a szövegek előfeldolgozásáért, az algoritmusok futtatásáért és a vizuális visszajelzésért. A komponensek egymással és a felhasználói bevitellel összhangban működnek, elősegítve az interaktív tanulási folyamatot.

Tesztelés

Felhasználói történetek

AS A		Felhasználó
I WANT TO		Főoldal böngészése
1.	GIVEN	A felhasználó elérte a weboldalt.
	WHEN	Megjelenik a főoldal.
	THEN	Lát egy rövid áttekintést a három mintaillesztő algoritmusról, és választhat a sötét vagy világos mód között. Továbbá, lehetősége van angol vagy magyar nyelv kiválasztására.
I WANT TO		Sötét/Világos mód váltás
2.	GIVEN	A felhasználó bármely oldalon van.
	WHEN	Rányom a sötét/világos gombra
	THEN	Az oldal design-ja azonnal átáll a kiválasztott témára
I WANT TO		Nyelv változtatás
3.	GIVEN	A felhasználó bármely oldalon van.
	WHEN	Rányom, az angol/magyar gombra
	THEN	Az oldal nyelve megváltozik a kiválasztottra
I WANT TO		Átmenni egy kiválasztott oldalra
4.	GIVEN	A felhasználó bármely oldalon van.
	WHEN	Rányom a kiválasztott oldalra

	THEN	Megjelenik a kiválasztott oldal
I WANT TO		Tanulni az egyszerű mintaillesztés algoritmusról
5.	GIVEN	A felhasználó az egyszerű mintaillesztési oldalon van.
	WHEN	Saját szöveget és mintát ad meg, majd elindítja az algoritmust.
	THEN	A szimuláció során láthatja, ahogy az algoritmus a megadott mintát a szövegben keresi, és megértheti a működését.
I WANT TO		Tanulni a QuickSearch algoritmusról
6.	GIVEN	A felhasználó a QuickSearch oldalon van.
	WHEN	Saját szöveget és mintát ad meg, majd elindítja az algoritmust.
	THEN	A szimuláció előtt megismerheti a shift-értékek táblázatát, majd láthatja az algoritmus lépéseit, és hogyan találja meg a mintát a szövegben.
I WANT TO		Tanulni a KMP algoritmusról
7.	GIVEN	A felhasználó a KMP oldalon van.
	WHEN	Saját szöveget és mintát ad meg, majd elindítja az algoritmust.
	THEN	A szimuláció előtt megismerheti a π- függvényt, majd láthatja az algoritmus lépéseit, és hogyan találja meg a mintát a szövegben.
I WANT TO		Leállítani a szimulációt

8.	GIVEN	A felhasználó bármely oldalon van, és a szimuláció fut.
	WHEN	Rányom a leállítás gombra.
	THEN	A szimuláció azonnal leáll.
I WANT TO		Újrakezdeni a szimulációt
9.	GIVEN	A felhasználó bármely oldalon van, és a szimuláció fut/futott már.
	WHEN	Rányom az újrakezdés gombra.
	THEN	A szimuláció visszaáll az eredeti állapotba, és a felhasználó újra kezdheti a gyakorlást.
I WANT TO		Megismerni az algoritmusok működését
10.	GIVEN	A felhasználó bármely oldalon van (kivéve kezdőlap).
	WHEN	Az algoritmus ismertetés gombra kattint.
	THEN	Megjelenik részletes leírás az adott oldalon alkalmazott algoritmusról, segítve a felhasználót a jobb megértésben.
I WANT TO		Manuálisan irányítani a szimulációt
11.	GIVEN	A felhasználó bármely oldalon van (kivéve kezdőlap).
	WHEN	A manuális irányítás gombra kattint.
	THEN	Megjelenik a következő lépés gomb.
12.	GIVEN	Megjelent a következő lépés gomb.

	WHEN	Rákattint a következő lépés gombra.
	THEN	Az algoritmus egy lépést bemutatott.
13.	GIVEN	Legalább egy lépést végrehajtott az algoritmus.
	WHEN	Rákattint az előző lépés gombra.
	THEN	Az algoritmus visszalépett az előző lépésbe.
13.	GIVEN	Legalább egy lépést végrehajtott az algoritmus.
	WHEN	Rákattint az előkészítés gombra.
	THEN	Az algoritmus visszalép a kezdeti állapotba.
I WANT TO		Gyakorolni a mintaillesztő algoritmusokkal kapcsolatos tudásomat
14.	GIVEN	A felhasználó megnyitja a gyakorló oldalt
	WHEN	Megjelennek a kérdések és válaszlehetőségek
	THEN	A felhasználó egyértelműen láthatja, hogy mit kell tennie, és megkezdheti a kvízt
I WANT TO		Válaszolni a kérdésekre
15.	GIVEN	Egy kérdés és annak válaszlehetőségei megjelentek
	WHEN	A felhasználó rákattint egy válaszlehetőségre
	THEN	A rendszer automatikusan továbblép a következő kérdésre
I WANT TO		Megtudni, hogy melyik válaszom volt helyes vagy hibás

14.	GIVEN	A kvíz befejezése után a felhasználó a „Válaszok megtekintése” gombra kattintott.
	WHEN	Megjelennek az összes válasz és azok eredményei
	THEN	A helyes válaszok zöld, a hibásak piros színnel vannak kiemelve

Manuális tesztelési terv

Kezdőoldal Tesztelése

1. Kezdőoldal megjelenítése
   Nyissa meg a weboldalt a böngészőben.
   Ellenőrizze, hogy a kezdőoldal betöltődik-e.
   Ellenőrizze a sötét/világos mód váltási lehetőséget.
   Ellenőrizze az angol/magyar nyelvváltási lehetőséget.

Szimulációt bemutató oldalak Tesztelése

2. Nyissa meg a megfelelő algoritmus oldalát ("Egyszerű mintaillesztés", "QuickSearch", "KMP").

   Ellenőrizze, hogy az oldal helyesen betöltődik.

3. Kattintson az "Indítás" gombra, figyelje meg a szimulációt, és ellenőrizze, hogy helyes eredményt ad-e.
4. Adjon meg saját szöveget és mintát, majd ismét indítsa el a szimulációt, és ellenőrizze az eredményt.
5. Kattintson a "Manuális irányítás" gombra, és kövesse végig a szimuláció lépéseit a "Következő lépés" és "Előző lépés" gombokkal.
6. QuickSearch oldalán: Ellenőrizze, hogy a shift-értékek táblázata helyesen megjelenik.
7. KMP oldalán: Ellenőrizze, hogy a π (pi) függvény számítása helyesen történik.

Gyakorló oldal Tesztelése

8. Válaszadás működése

   Kattintson bármely válaszra, ellenőrizze, hogy a kvíz automatikusan továbblép-e a következő kérdésre.

9. Kvíz eredménye

Kattintson a „Válaszok megtekintése” gombra, ellenőrizze, hogy a kiválasztott válaszok láthatók. A helyes válaszok zöld, a hibás válaszok piros színnel jelenjenek meg.

Sötét/Világos módváltás Tesztelése

10. Nyelvváltás tesztelése

    Váltson angol nyelvre, és ellenőrizze, hogy minden szöveg megfelelően megjelenik-e angolul.

    Váltson vissza magyar nyelvre, és ellenőrizze, hogy minden szöveg visszaáll-e magyarra.

11. Sötét/világos mód tesztelése

    Váltson sötét módra, és ellenőrizze, hogy az oldal helyesen jelenik-e meg.

    Váltson vissza világos módra, és ellenőrizze, hogy az oldal visszaáll-e az eredeti állapotába.

Automatizált tesztelés

Angular tesztelési infrastruktúrája

Az Angular tesztelési megoldása egy robusztus és jól dokumentált infrastruktúrát biztosít a fejlesztők számára. Az egyik kulcsfontosságú elem ebben a kontextusban a TestBed, amely az Angular DI (Dependency Injection) rendszerére építve létrehozza és konfigurálja a tesztkörnyezetet. A TestBed segítségével beállíthatjuk a szükséges modulokat, deklarációkat, illetve szimulálhatjuk a komponens élettartamát, így biztosítva, hogy a komponensek pontosan úgy működjenek, mint a valós alkalmazásban.

A dolgozatban szereplő példák között szerepelnek olyan alapvető, egységteszt jellegű esetek is, ahol a komponensek létrehozásától kezdve az egyes metódusok – mint például a szöveg feldolgozása, mintaillesztési lépések vagy a vizuális kijelölések kezelése – működését validáljuk. Az aszinkron viselkedést a fakeAsync és tick funkciók alkalmazásával teszteljük, amelyek lehetővé teszik a valós idejű függőségek szinkron módon történő vizsgálatát.

Tesztelési típusok és módszertan

A dolgozat során az alábbi tesztelési típusokat használtam, amelyek együttesen biztosítják a rendszer megbízhatóságát:

Egységtesztek

-Célja: Az egyes komponensek, függvények és szolgáltatások izolált működésének ellenőrzése.

-Módszer: A komponensek inicializálása a TestBed segítségével, majd az adott metódusok futtatása és a kimenetek összehasonlítása az elvárt értékekkel. Például, ellenőrizhető, hogy a komponens megfelelően állítja be a megjelenítendő szöveget és más belső állapotváltozókat a megadott bemenetek alapján.

Integrációs tesztek

-Célja: Több egységet összekötő komponens(ek) együttműködésének vizsgálata.

-Módszer: Bár az egységtesztek izoláltan vizsgálták a komponensek működését, az integrációs tesztek során ellenőrizni tudjuk, hogy a különböző részek hogyan lépnek interakcióba egymással. Ez különösen fontos olyan alkalmazások esetén, ahol a komponensek egymásra épülnek, és a hibák az összekapcsolódás során jelentkeznek.

Aszinkron tesztek

-Célja: Az időfüggő műveletek – például animációk, időzített események vagy szerverrel történő kommunikáció – szinkron tesztelése.

-Módszer: A fakeAsync és tick funkciók segítségével lemodellezhető a valós idejű viselkedés, így az aszinkron függőségek determinisztikusan és előre jelezhetően leteszthetők.

Ezek a módszertanok együttesen lehetővé teszik, hogy a kód minden egyes módosítása után a fejlesztői csapat gyors visszajelzést kapjon, így az esetleges regressziókat már azonosítás után javítani tudják.

Tesztek futtatása és automatizációja

Az Angular CLI által biztosított „ng test” parancs segítségével a tesztelési folyamat egyszerűen integrálható a fejlesztési ciklusba. A parancs futtatása automatikusan elindítja a Karma tesztfuttatót, amely a konfigurációs fájlban megadott beállítások alapján keresi a *.spec.ts kiterjesztésű fájlokat, majd végrehajtja a tesztek sorozatát. A futtatás eredménye részletesen megjelenik a konzolon, és a generált jelentések segítségével a fejlesztők könnyen nyomon követhetik az esetleges hibákat.

Az automatizált tesztelés különösen hasznos a folyamatos integráció (CI) környezetekben, ahol minden kódváltoztatást azonnal le kell tesztelni. Ez a megközelítés biztosítja, hogy a kód

bármilyen módosítása esetén a meglévő funkciók továbbra is hibátlanul működjenek, így a rendszer stabilitása és a minőségi követelmények folyamatosan garantáltak.

Tesztelés eredményei

A végrehajtott manuális és automatizált tesztelések alapján a rendszer funkciói a várakozásoknak megfelelően működtek. A manuális tesztek során a felhasználói történetekben megfogalmazott elvárások teljesültek: az oldalak helyesen betöltődtek, a navigáció, az algoritmusok léptetése, valamint a sötét/világos mód és a nyelvváltás funkciók megfelelően reagáltak.

19. tesztelés eredménye

Az automatizált egység- és integrációs tesztek során 84 teszteset futott le sikeresen hiba nélkül, amit a mellékelt ábra is igazol (19. ábra).

A tesztelés összességében igazolta, hogy a rendszer stabilan működik a megadott funkcionális követelmények mellett.

Fejlesztési lehetőségek

A "Mintaillesztő algoritmusok szemléltetése" webalkalmazásom számos továbbfejlesztési lehetőséget kínál, amelyekkel a felhasználói élményt javíthatom, a funkcionalitást bővíthetem, és az alkalmazás hatékonyságát növelhetem. Az alábbiakban bemutatom az általam javasolt potenciális fejlesztési irányokat.

Új Algoritmusok Integrálása

A jelenlegi három algoritmus (Egyszerű mintaillesztés, QuickSearch, Knuth-Morris-Pratt) mellett további mintaillesztő algoritmusok beépítése növelheti az alkalmazás hasznosságát és oktatási értékét. Például:

- Boyer-Moore algoritmuscsalád további változatai: például a Boyer-Moore-Horspool vagy a Turbo Boyer-Moore algoritmusok
- Rabin-Karp algoritmus: Hash függvények használatával gyorsítja fel a mintaillesztést.

Teljesítmény Optimalizálása

Az algoritmusok futás idejének és hatékonyságának részletes mérése és optimalizálása nagyobb szövegeken végzett műveletek esetében is hasznos lehet. Ezzel a felhasználók valós idejű visszajelzést kaphatnak az algoritmusok teljesítményéről.

Felhasználói Felület Javítása

A felhasználói felület további finomítása és egyszerűsítése javíthatja az alkalmazás használhatóságát. Ez magában foglalhatja:

- Reszponzív design: Az alkalmazás optimalizálása különböző eszközökön, például mobiltelefonokon és tableteken való használatra.
- Jobb vizuális visszajelzés: Az algoritmusok futása közben történő valós idejű visszajelzés javítása, például haladásjelzők és részletesebb eredménybemutatás.

Fejlett Eredmény Elemzés

Az algoritmusok eredményeinek részletesebb elemzése és vizualizálása segíthet a felhasználóknak mélyebb megértést nyerni a mintaillesztési folyamatokról. Ez magában foglalhatja:

- Részletes statisztikák: Az algoritmusok teljesítményének statisztikai elemzése, például futási idő és karakter-összehasonlítások száma.
- Összehasonlító elemzés: Különböző algoritmusok eredményeinek és teljesítményének összehasonlítása egy közös nézetben.

Ezek a továbbfejlesztési lehetőségek nemcsak az alkalmazás funkcionalitását bővíthetik, hanem hozzájárulhatnak annak oktatási értékéhez is, valamint javíthatják a felhasználói élményt. Az alkalmazás jövőbeni fejlesztései során ezekre a területekre fogok koncentrálni, hogy a lehető legjobb felhasználói élményt nyújtsam.