Fejlesztői dokumentáció

Gameknowledge

Tartalomjegyzék

# Bevezetés

A Gameknowledge című weboldal annak céljából készült, hogy a felhasználók könnyedén tájékozódhassanak a videójáték ipar fejlődéseiről, változásairól, mindezek mellett bárki egyszerűen találhasson információkat kedvenc, de akár régebbi játékairól egy helyen, összeszedett és átlátható formában.

A Gameknowledge fő célja, hogy a rengeteg információt, amely kering az interneten a játékokról egyhelyre gyűjtve tiszta, hiteles valamint gördülékenyen visszakereshető formában tárolja és jelenítse meg a felhasználóknak.

Az oldal bizonyos szolgáltatásait bárki igénybe veheti minden gond nélkül, azonban bizonyos funkciókat a felhasználók csak az oldalra való sikeres regisztráció után használhatnak. Ezen funkciók megtalálhatóak a Használati útmutató részben.

Verziószám: 0.2.2

Kapcsolattartók:

Földházi Zsolt,

[foldhazi.zsolt.2019@osztalyterem.katolikuskeri.hu](mailto:foldhazi.zsolt.2019@osztalyterem.katolikuskeri.hu)

Vicickó Vencel,

[vicicko.vencel.2019@osztalyterem.katolikuskeri.hu](mailto:vicicko.vencel.2019@osztalyterem.katolikuskeri.hu)

# Rendszerkövetelmények

Bármely webböngészőt futtatni képes operációs rendszeren képes működni a weboldal.

Egy legalább Intel Pentium 4-es processzor vagy későbbi, amely képes az SSE3-ra elegendő hardware a működéshez, azonban ajánlott egy Intel(R) Core(TM) i3-1005G1 1.20GHz processzor, 4 GB RAM valamint egy Windows 11-es operációs rendszer.

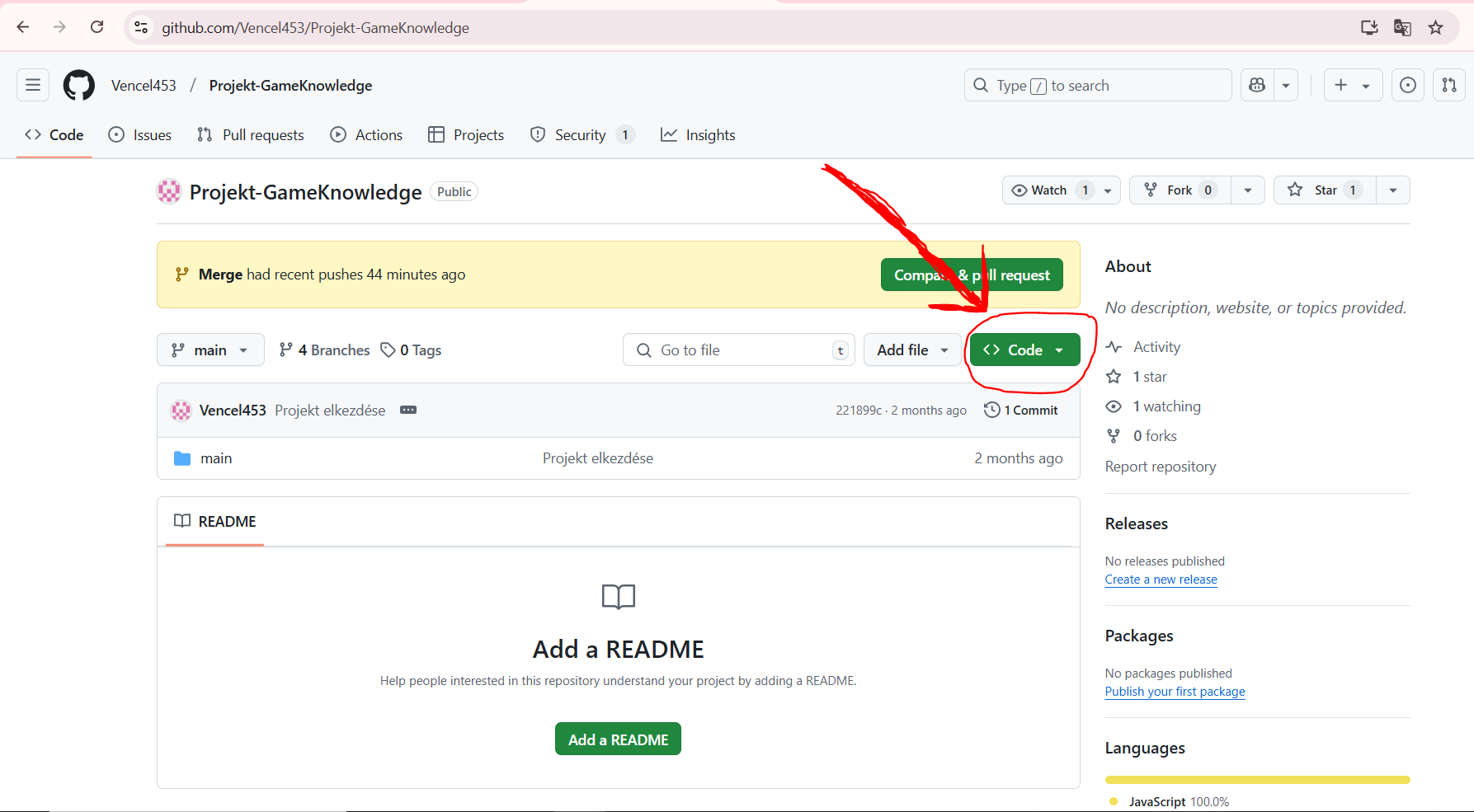
Mindezeken felül, az oldal működéséhez elengedhetetlen a game\_knowledge adatbázis, az Angular keretrendszer 18-as verziója és a NodeJS szoftverrendszer.

# Telepítési útmutató

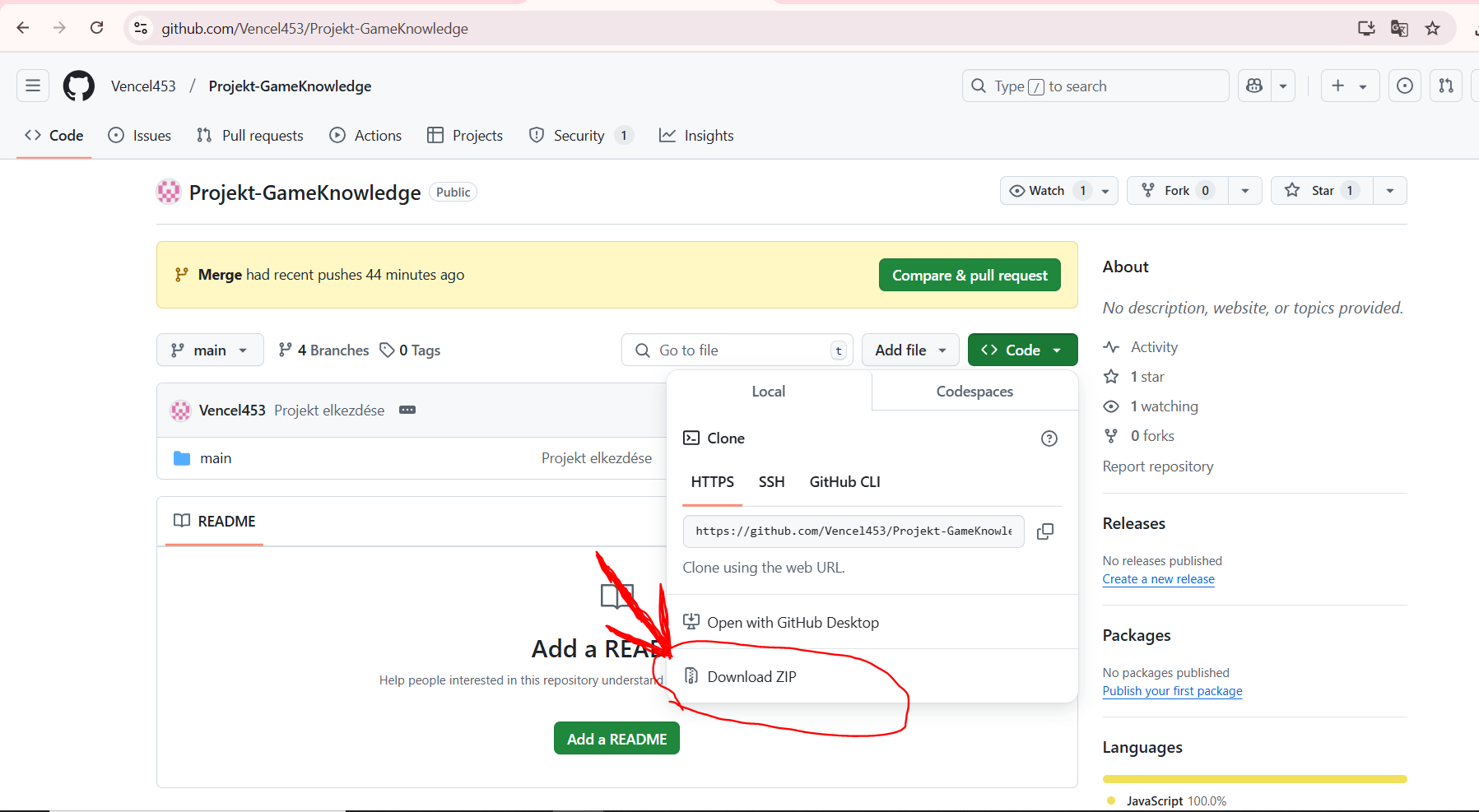
1. Szükség van a forrás kódokra, amelyeket a Github weblapunkról egyszerűen letölthetőek. Navigáljon az oldalunkra az alábbi link segítségével és kövesse a további lépéseket!

<https://github.com/Vencel453/Projekt-GameKnowledge.git>

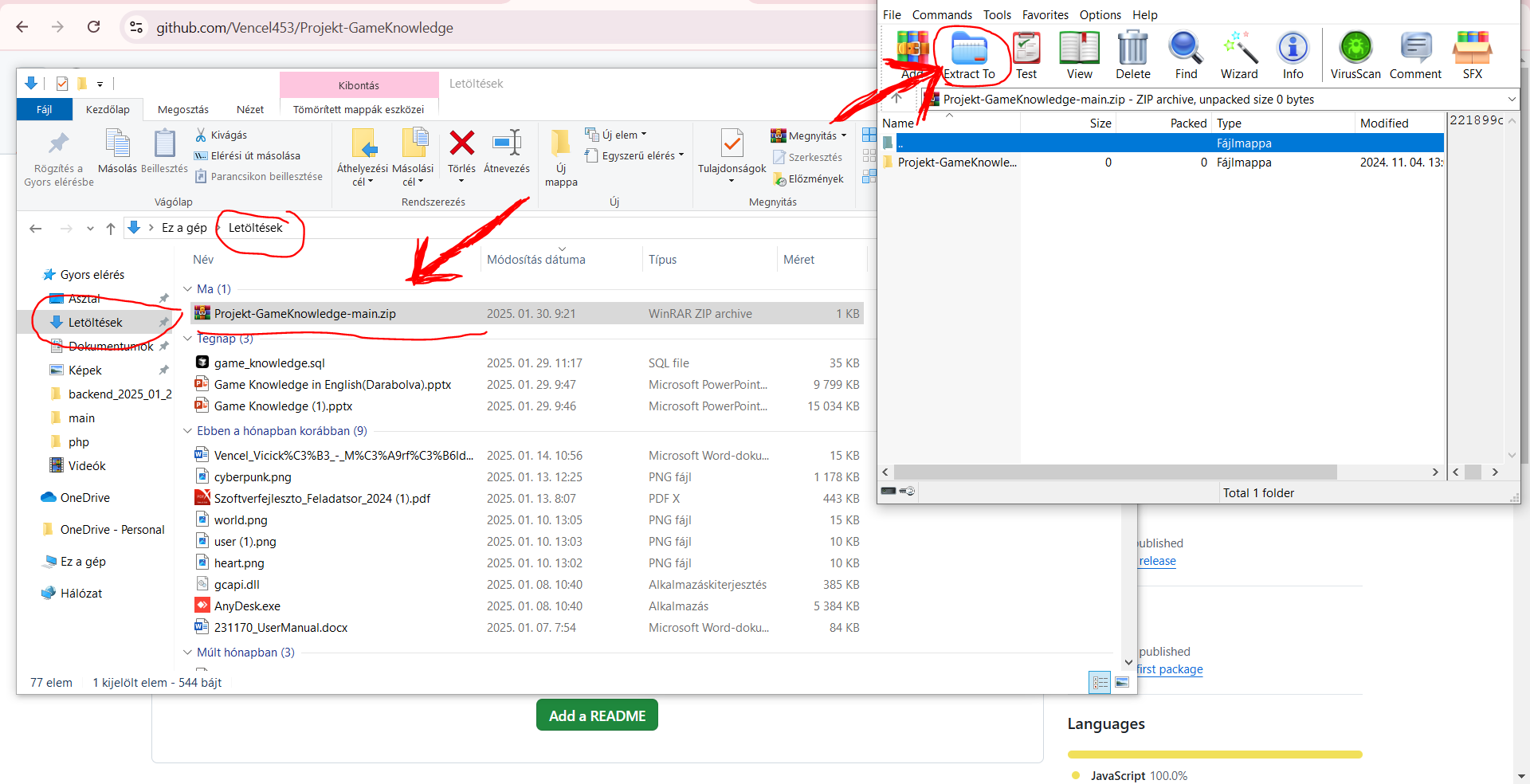
1. Az alábbi kép alapján válassza a < >Code nevű lenyíló menü-t!



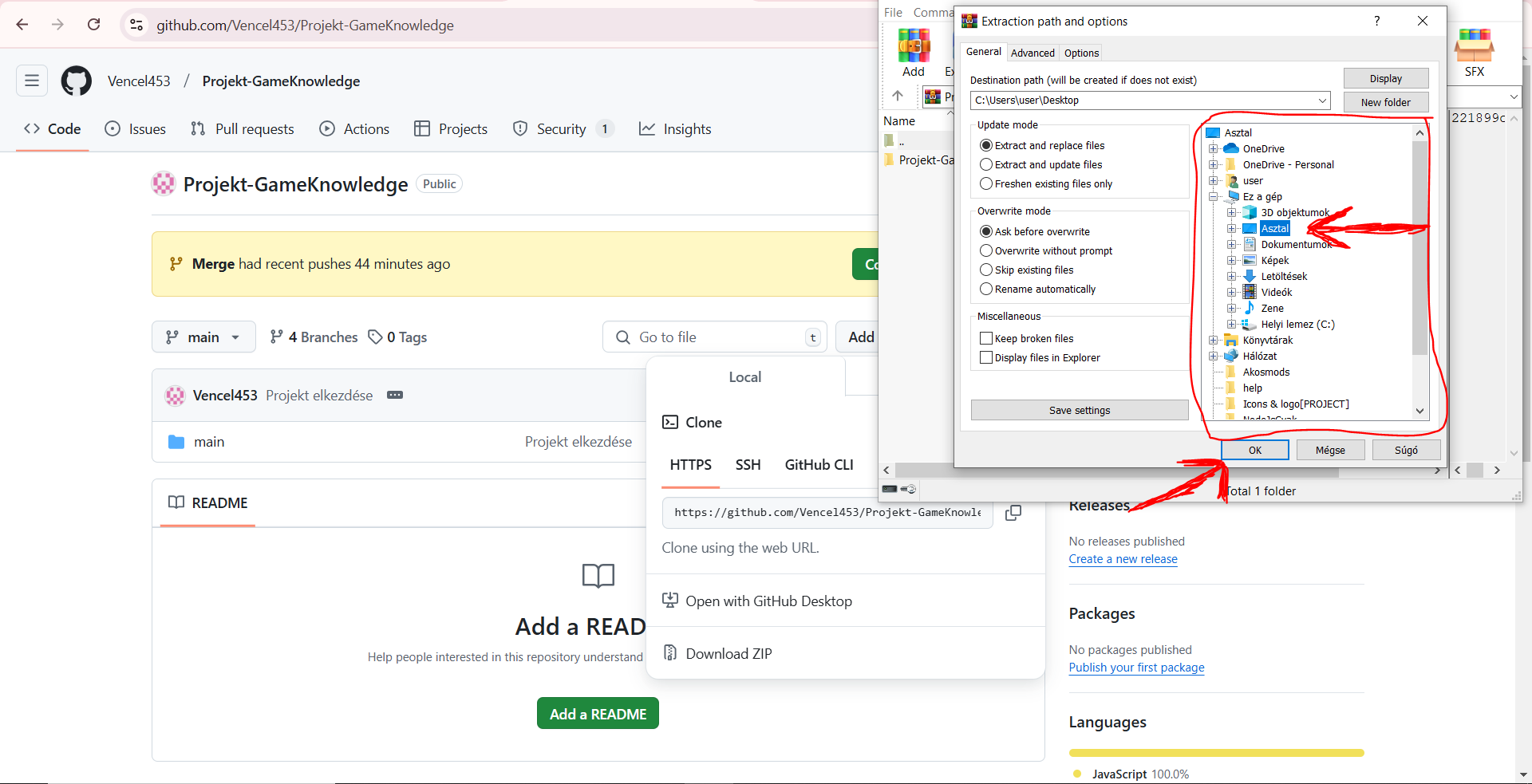
1. *kép*
2. Az alábbi kép alapján válassza a Download ZIP opciót és töltse le a tömörített állományt!



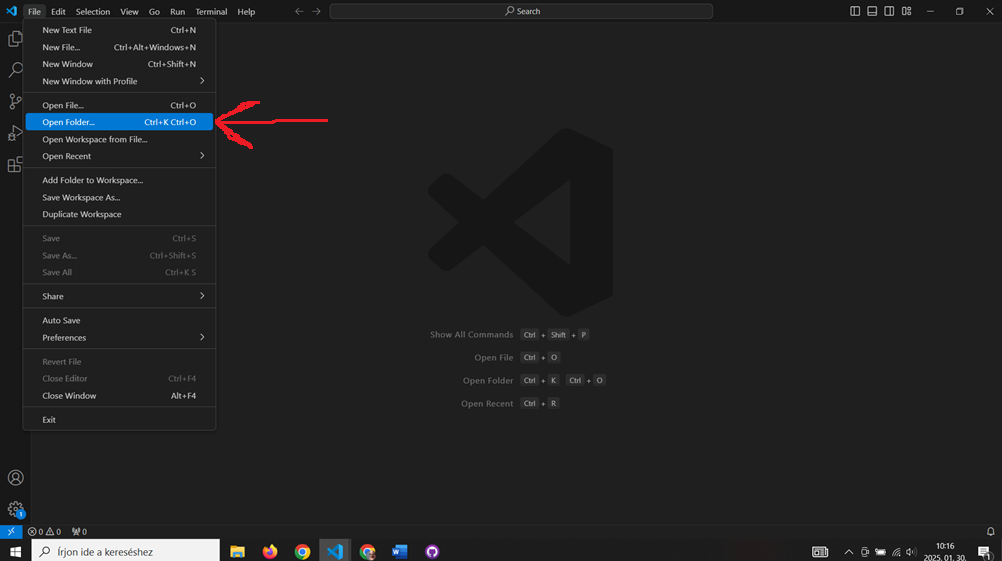
1. *kép*
2. Keresd meg a letöltött ZIP állományt a saját gépeden (valószínűleg a letöltésekben lesz megtalálható), és nyisd meg valamelyik tömörítő programmal, majd az általad kiválasztott helyre bontsd ki!



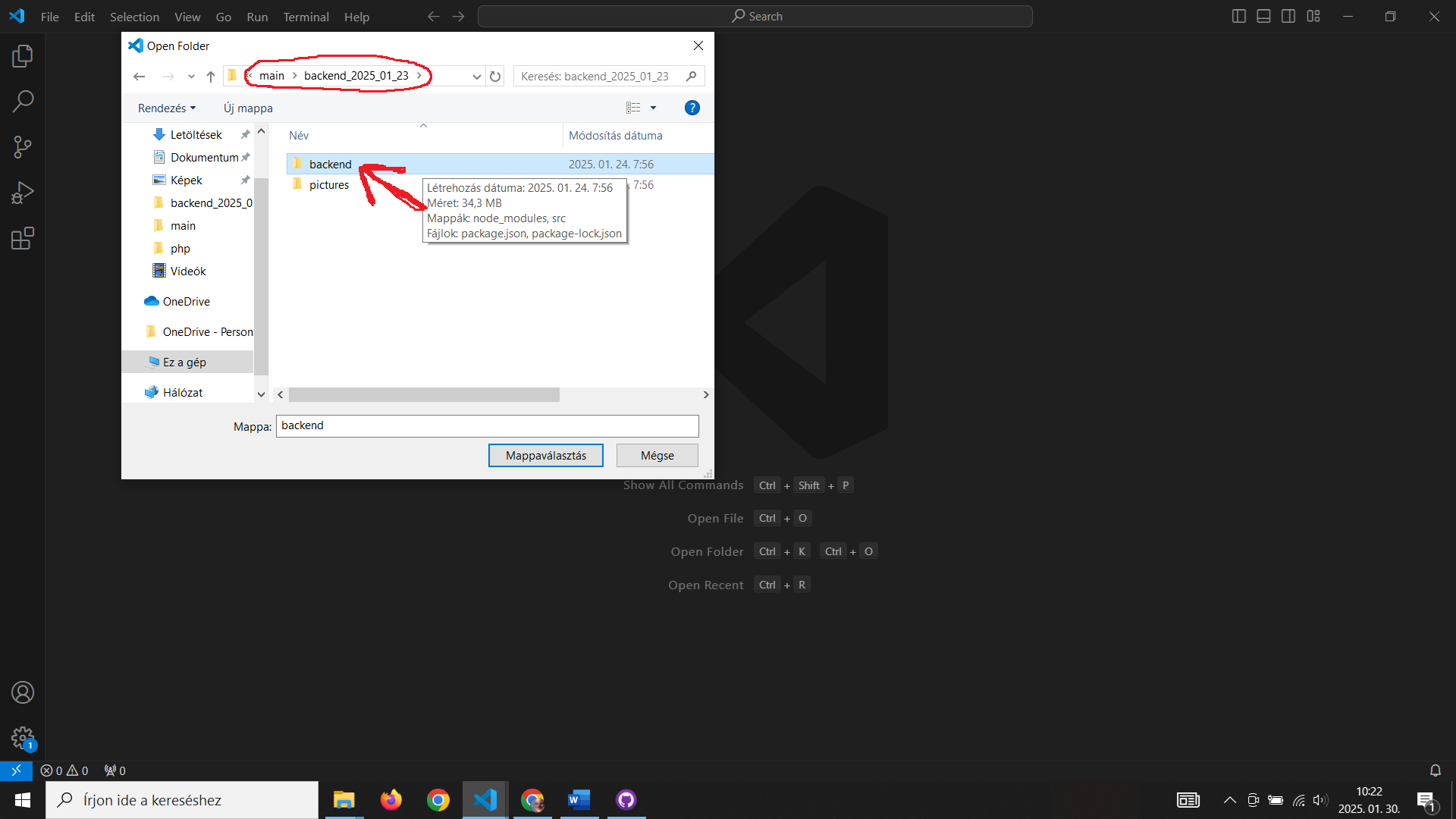
1. *kép*
2. A felugró ablak jobb oldalán válaszd ki, hogy hova szeretné kicsomagolni a fájlt (az alábbi kép jelenleg az Asztalt válassza), majd kattintson az OK gombra!



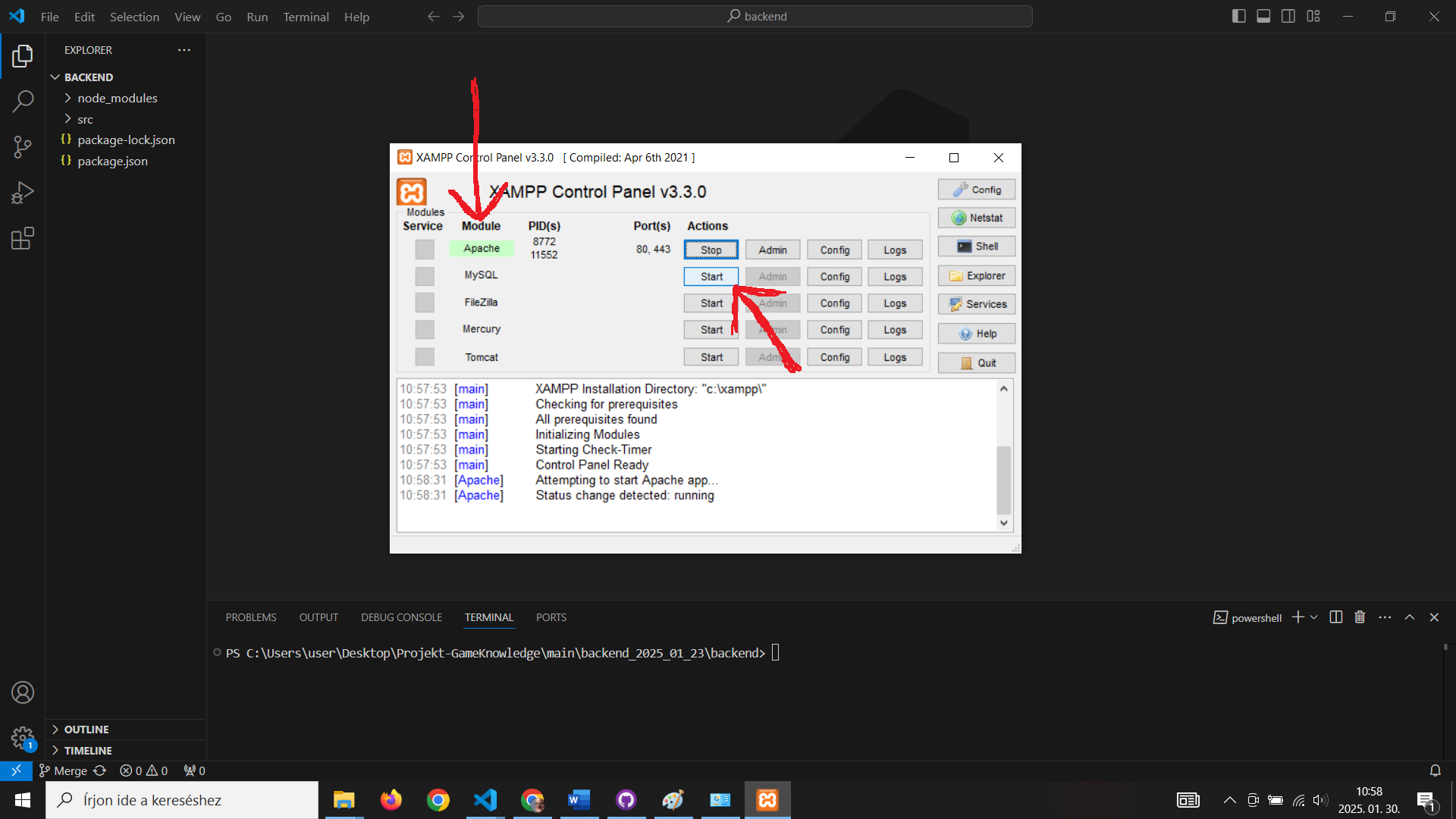
1. *kép*
2. Nyissa meg a Visual Studio Code programot és válassza ki az alábbi opciót az alábbi kép alapján!



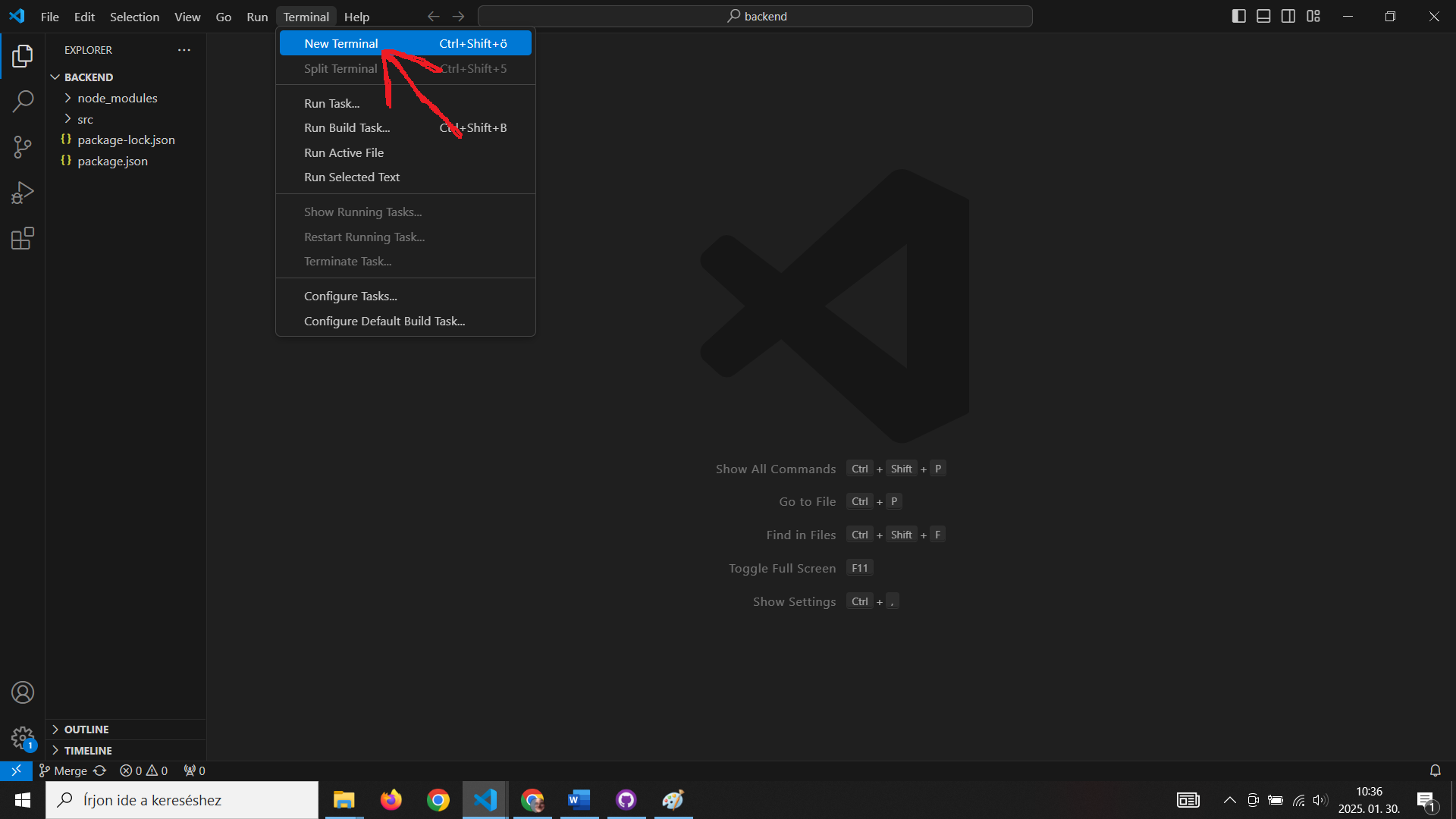
1. *kép*
2. Válassza ki a kicsomagolt mappát, majd azon belül is a backend mappát és nyissa meg a Visual Studio Code-ban!



1. *kép*
2. Nyissa meg a XAMPP nevű programot, és indítsa el az Apache és MySQL modulokat az alábbi kép alapján!

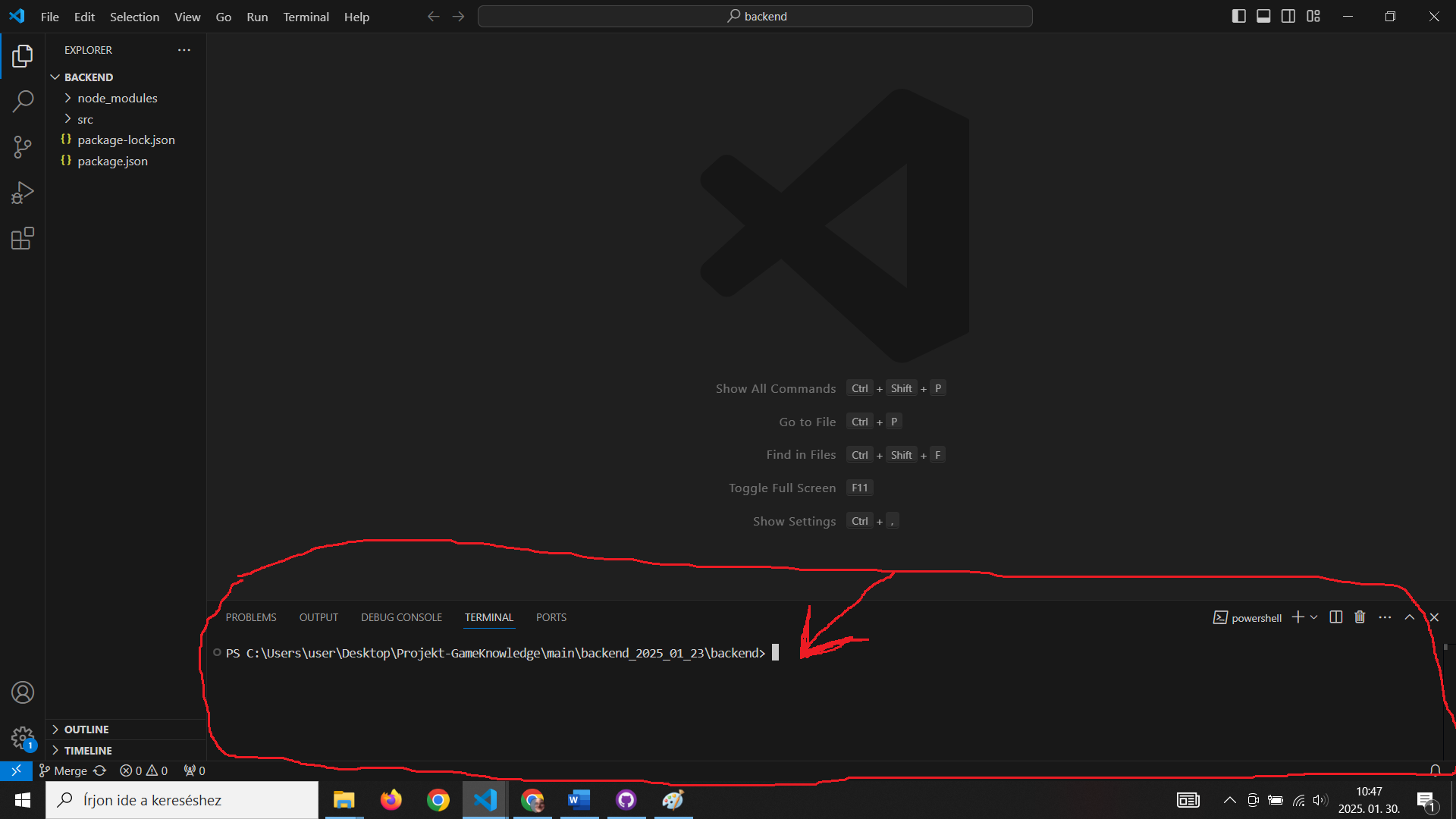


1. *kép*
2. Válassza ki a Terminál opciót az alábbi kép alapján, majd ezen belül is a New Terminal opciót!



1. *kép*
2. Az alul megjelenő terminálba írja be a következő parancsot: npm run dev

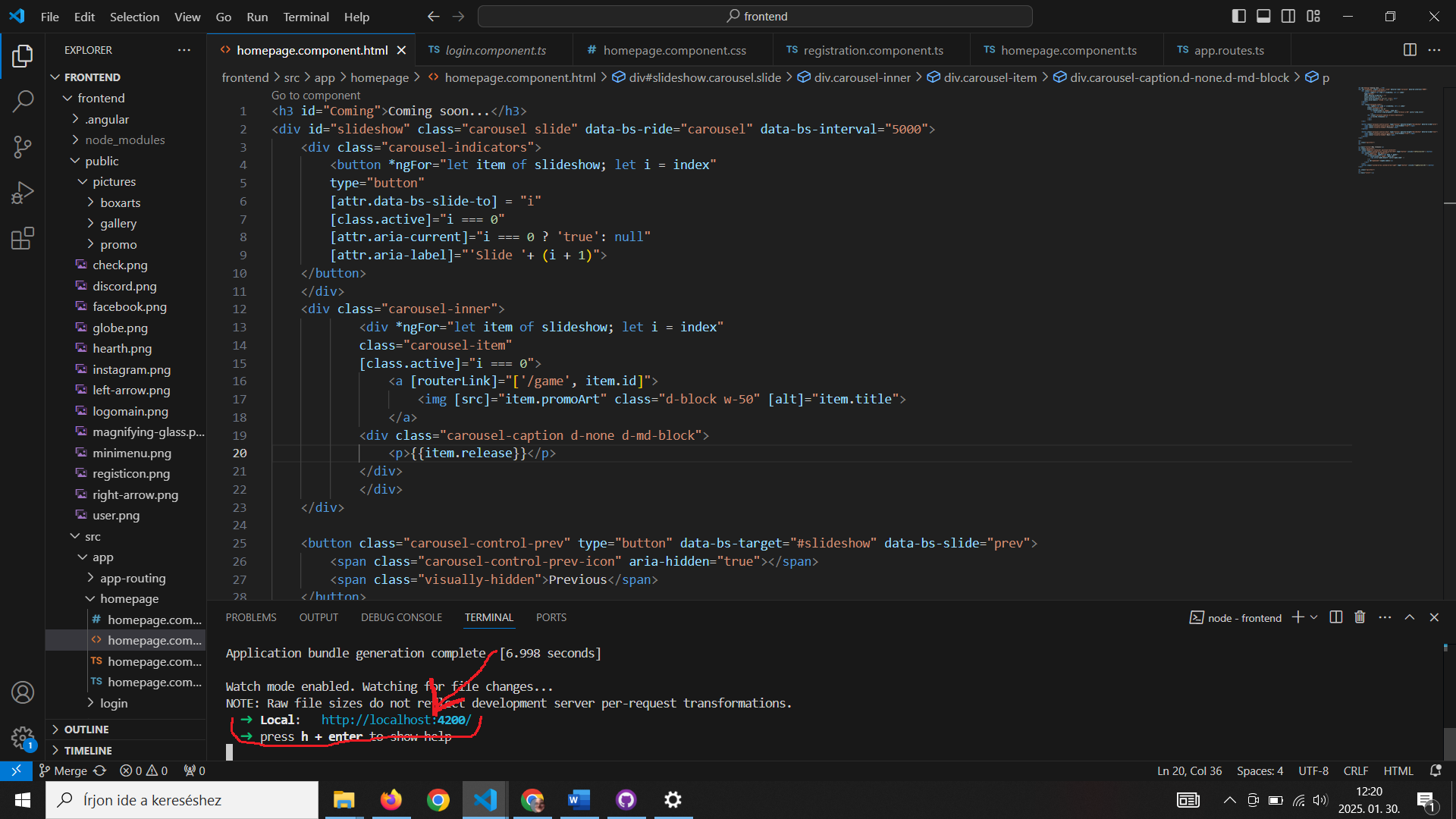
Ezzel elindítja a backend szervert, amely kapcsolatban áll az adatbázissal, és a weblap műveleteit fogja végezni.



1. *kép*
2. A backend szerver leállítása a CTRL + C billentyű kombináció leütésével állítható le.
3. Az előzőekben letöltött és kicsomagolt állományból (lásd a *3. képen*), válassza ki és nyissa meg Visual Studio Code-ban a frontend mappában található frontend állományt (lásd a *6. képen*)!
4. Válassza ki ismét a Terminal, azon belül is a New Terminal opciót (lásd a *8. képen*)!
5. Az alul megjelent terminálban (lásd a *9. képet*) gépelje be a következő parancsot: ng serve

Ezáltal elindul localhost-on az angular szervere.

1. Kattintson az alábbi képen ábrázolt localhost-os linkre, amely megjelent a terminálban, és a weboldal megjelenik az ön által alapértelmezetten használt böngészőjében.



1. *kép*

# Frontend fejlesztői dokumentáció

## Általános információk

A weboldal frontend része az Angular 18-as verziójában készült, amely keretrendszer azért lett kiválasztva mert biztosítani tudja számunkra számos funkciókat (pl.: a weboldal localhost-on való futtatását vagy a felmerülő kritikus hibák esetén a nagyobb visszalépések lehetőségét). A keretrendszerben megalkotott frontend tartalmaz bootstrap elemeket és osztályazonosítókat valamint statikus képi és ikon fájlokat. Mindezek mellett a weboldal frontend része úgynevezett standalone komponensekből épül fel, amelyek összessége alkotja az oldal(ak) felépítéseit, tartalmait, funkcióit és stílusait.

## Mappastruktúra

.vscode

public

src

├── app

│ ├── app-routing

│ ├── homepage

│ ├── login

│ │ ├── login.component.css

│ │ ├── login.component.html

│ │ ├── login.component.spec.ts

│ │ ├── login.component.ts

│ ├── page-not-found

│ │ ├── page-not-found.component.css

│ │ ├── page-not-found.component.html

│ │ ├── page-not-found.component.spec.ts

│ │ ├── page-not-found.component.ts

│ ├── registration

│ │ ├── registration.component.css

│ │ ├── registration.component.html

│ │ ├── registration.component.spec.ts

│ │ ├── registration.component.ts

│ ├── specificgame

│ ├── app.component.css

│ ├── app.component.html

│ ├── app.component.spec.ts

│ ├── app.component.ts

│ ├── app.config.ts

│ ├── app.config.server.ts

│ ├── app.routes.ts

├── index.html

├── main.server.ts

├── main.ts

├── styles.css

.editorconfig

.gitignore

angular.json

package-lock.json

package.json

README.md

server.ts

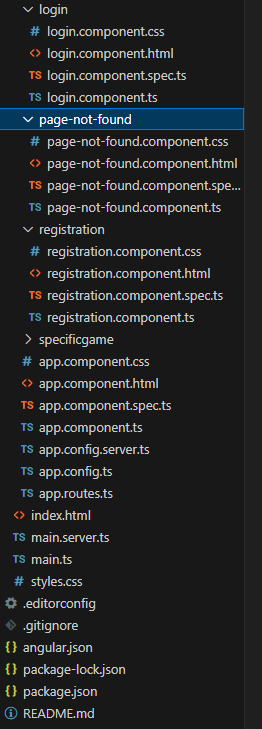
tsconfig.app.json

tsconfig.json

tsconfig.spec.json

A fentiekben a frontend mappastruktúrája látható. Fentről lefelé haladva a public nevű mappa tartalmazza a statikus képeket amelyeket használ a frontend egyes helyeken (pl.: navigációs sáv kereső nagyító ikonja). Ezt követően az src nevű mappa, amely tartalmazza a főbb fájlokat. Ezek többek közt a standalone komponensek, mint a homepage, a login vagy a registration, majd következnek az app. fájlok mint az app.component.html, az app.component.ts, vagy az app.component.css. Ezekben a fájlokban találhatóak azok a weboldal elemek, stílusok és funkciók amelyek az összes más különálló (standalone) komponensekre érvényesek kell hogy legyenek. Példának okáért ilyen a navigációs sáv, annak tartalmai és funkciói amelyeknek meg kell jelenniük és működniük kell az összes oldalon. Az ezeket követő fájlok (angular.json, package.lock.json stb.) már az angular megfelelő működéséért felelnek. Ezek óvatlan módosításai, vagy eltávolításuk kritikus hibákat eredményezhetnek!

## Standalone komponens alapú felépülés



Ebben a részben a weblap standalone, avagy különálló komponenseiről olvashat. A fenti képen láthat egy részletet a mappastruktúrából. A komponensek a következők: login, és a hozzá tartozó többi fájl, a page-not-found, a registration, az app. fájlok az app.routes.ts kivételével, valamint a specificgame. Ezen komponensnek tartalmazzák a weblap oldalainak megfelelő tartalmát, stílusát és funkcióját. A login komponens a bejelentkező felületet valósítja meg, a registration a regisztrációs felületért felel, a page-not-found amikor valamilyen oknál kifolyólag egy adott oldal nem elérhető így alapértelmezetten átirányításra kerül erre az oldalra, és a specificgame komponens amely mindig, a felhasználó által kiválasztott specifikus játék oldalát valósítja meg. Ezen komponensek függetlenek egymástól, mégis képesek együtt dolgozni. Egy átláthatóbb megoldás hosszútávon, azonban ha a későbbiekben szükséges módosítások megkövetelik valamilyen modul, más komponens vagy csomag importálását, akkor tudni kell, hogy mely más komponenseknek is szüksége van e az újonnan importált dolgok valamelyikére, és ha igen akkor azt minden egyes komponensben importálni szükséges. A komponensnek 4db fájlt tartamaznak: HTML, CSS és 2 Typescript fájlt. Esetleges módosítások esetén önnek csak a .spec.ts-re végződő fájl kivételével bármelyiket módosíthatja, ugyanis a 4 fájl közül csak 3 az, amelyik a komponens tartalmának manipulálására szolgál, a .spec.ts-re végződő fájl egy alapvető konfigurációt tartalmaz melyet az angular hoz létre a komponens létrehozásakor annak céljából, hogy a komponens megfelelően működjön.