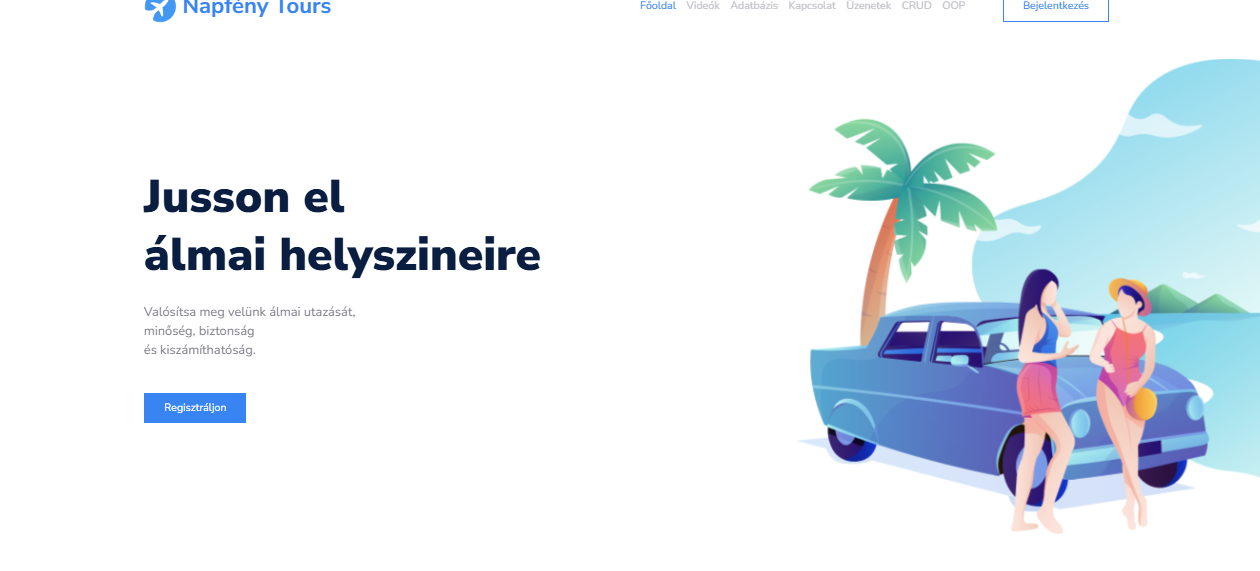
****

**WEB-PROGRAMOZÁS II – Levelező – Beadandó feladat – NodeJS   
  
Utazás   
  
  
weboldalt késztette:**

**Czentye Györgyi Tímea AX9NJQ és  
Fekete Pál PIXUNC**

**A feladat leírása:**

Egy Node.js-alapú webes alkalmazást kell készíteni egy adott Linux szerveren. Az alkalmazás egy kiválasztott adatbázis adatait fogja megjeleníteni és kezelni. A feladat során fontos a csapatmunka, a verziókövetés (Git) és a dokumentáció.

A feladat részletei:

* Csapatmunka: Két fős csoportokban kell dolgozni, a feladatokat megosztva.
* Adatbázis: Egy adott Google Drive mappában található adatbázisok közül kell választani.
* Funkciók:
  + Főoldal: A cég bemutatása.
  + Adatbázis: Az adatbázis adatainak megjelenítése.
  + Kapcsolat: Üzenetküldő űrlap.
  + Üzenetek: Elküldött üzenetek listázása.
  + CRUD: Adatok létrehozása, olvasása, módosítása és törlése.
  + OOP-JavaScript: Objektum-orientált JavaScript alkalmazás.
* Technológiák: Node.js, HTML, CSS, JavaScript, SQL
* Egyéb: Git verziókövetés, dokumentáció

**A feladat megoldásának javasolt sorrendje**

1. Csapatmunka szervezése:
   * Feladatmegosztás: Ki melyik részt készíti el?
   * Kommunikáció: Milyen eszközöket használnak a kommunikációhoz (pl. Teams, Discord)?
   * Verziókövetés: Hogyan fogják használni a Git-et?
2. Adatbázis kiválasztása és megismerése:
   * Az adatbázis szerkezetének tanulmányozása.
   * A szükséges táblák és mezők azonosítása.
3. Szerverhez való csatlakozás és fejlesztői környezet beállítása:
   * SSH-kapcsolat létrehozása.
   * Node.js és egyéb szükséges eszközök telepítése.
4. Projektstruktúra felépítése:
   * Fájlstruktúra kialakítása.
   * Routing beállítása.
5. Főoldal elkészítése:
   * Reszponzív téma kiválasztása és beállítása.
   * Tartalom hozzáadása.
6. Adatbázishoz való kapcsolódás és adatok lekérdezése:
   * Adatbázis-kapcsolat létrehozása Node.js-ből.
   * SQL lekérdezések írása.
7. Adatok megjelenítése a weboldalon:
   * Lekérdezett adatok megjelenítése megfelelő formátumban (táblázatok, listák).
8. Kapcsolat űrlap és üzenetmentés:
   * Űrlap létrehozása HTML-ben.
   * Adatok küldése a szerverre.
   * Adatok mentése az adatbázisba.
9. CRUD műveletek implementálása:
   * Form-ok létrehozása új rekordok hozzáadásához.
   * Táblázatok létrehozása meglévő adatok megjelenítéséhez.
   * Módosítás és törlés funkciók implementálása.
10. OOP-JavaScript alkalmazás készítése:

* Egy egyszerű alkalmazás (pl. kalkulátor, todo lista) létrehozása objektum-orientált elvek alapján.

1. Git használat:

* Rendszeres commitok készítése.
* Pull request-ek létrehozása.
* Merge-elés.

1. Dokumentáció készítése:

* A projekt minden részletének részletes leírása.
* Használt technológiák, kódrészletek, képernyőképek.

1. Tesztelés és hibajavítás:

* Az alkalmazás alapos tesztelése.
* Hibaelhárítás.

A feladat egy komplex webfejlesztési projektet ír le, amely Node.js-t, adatbázist és egy választott témát ötvöz. Az alábbiakban lépésről lépésre vázolom a megoldás menetét és a figyelembe veendő szempontokat:

**1. Technológiai Választások**

* **Node.js:** A feladat alapja, a szerver oldali logikát biztosítja.
* **Express.js:** Népszerű web framework a Node.js-hez, egyszerűsíti a web alkalmazások fejlesztését.
* **Adatbázis:** A feladat nem határozza meg, így választhatunk MySQL, PostgreSQL, MongoDB vagy más megfelelő adatbázist.
* **Templating engine:** EJS, Pug vagy más templating engine a HTML generáláshoz.
* **Verziókövető:** GitHub, a követelményeknek megfelelően.

**2. Projektstruktúra**

projekt/

├── package.json

├── public/

│ ├── css/

│ ├── img/

│ └── index.html

├── routes/

│ ├── index.js

│ ├── users.js

│ └── ...

├── views/

│ ├── index.ejs

│ ├── users.ejs

│ └── ...

├── models/

│ ├── user.js

│ └── ...

└── app.js

* **public:** Statikus fájlok (CSS, JS, képek)
* **routes:** Útvonalkezelők
* **views:** Sablonok
* **models:** Adatbázis modellek
* **app.js:** Fő alkalmazás fájl

**3. Fejlesztés**

* **Adatbázis tervezés:** Az adatbázis struktúrájának megtervezése a választott téma alapján.
* **Modellek létrehozása:** Az adatbázis táblákhoz tartozó modellek létrehozása a Node.js-ben.
* **Útvonalak definiálása:** Az egyes oldalakhoz tartozó útvonalak és a hozzájuk tartozó logika megírása.
* **Sablonok készítése:** A választott témát felhasználva a sablonok elkészítése.
* **CRUD műveletek:** A választott táblához tartozó CRUD műveletek implementálása.
* **Form feldolgozás:** A kapcsolatfelvételi űrlap adatainak feldolgozása és az adatbázisba mentése.
* **Üzenetek megjelenítése:** Az elküldött üzenetek lekérdezése és megjelenítése fordított időrendben.
* **OOP JavaScript:** Egy egyszerű objektumorientált alkalmazás létrehozása, pl. egy felhasználó objektum.
* **GitHub használat:** A projekt folyamatos commitolása a GitHubra, branch-ek használata a különböző fejlesztési fázisokhoz.

**4. Dokumentáció**

* **Technikai részletek:** Használt technológiák, döntések indoklása.
* **Adatbázis séma:** Az adatbázis táblák részletes leírása.
* **Kódrészletek:** Fontosabb kódrészletek bemutatása.
* **Felhasználói útmutató:** Az alkalmazás használatának lépésenkénti leírása.
* **GitHub link:** A projekt GitHub repository linkje.

**További Figyelnivalók**

* **Responzivitás:** A választott téma biztosítja a responzivitást, de érdemes további teszteket végezni különböző eszközökön.
* **Biztonság:** Az adatbázishoz való hozzáférés védelme, a felhasználói bevitelek szűrése.
* **Hibakezelés:** Hibakezelés implementálása az alkalmazásban.
* **Optimalizálás:** A teljesítmény optimalizálása, különösen nagyobb adatmennyiség esetén.
* **Stílusguide:** Egy következetes stílusguide használata a kód olvashatóságának javítása érdekében.

**Következő Lépések**

* **Téma kiválasztása:** A választott téma alapján határozzuk meg az adatbázis struktúráját és az alkalmazás funkcióit.
* **Fejlesztői környezet beállítása:** Node.js, adatbázis és egyéb eszközök telepítése és konfigurálása.
* **Projektstruktúra felépítése:** A fenti struktúra alapján a projekt létrehozása.
* **Iteratív fejlesztés:** A projekt fejlesztése kisebb lépésekben, folyamatos teszteléssel.
* **Dokumentáció készítése:** A fejlesztés során folyamatosan dokumentálni kell a munkát.

Szükséges Programok telepítése:

* **XAMPP:** Ez egy ingyenes, nyílt forráskódú fejlesztői környezet, amely tartalmazza az Apache web szervert, a MySQL adatbázist, a PHP nyelvet és a Perl nyelvet. Az XAMPP-t elsősorban a helyi gépeden történő fejlesztéshez használják, így könnyedén tesztelheted a weboldalad vagy alkalmazásod működését anélkül, hogy fel kellene töltened egy távoli szerverre.
* **Node.js:** Ez egy JavaScript runtime környezet, amely lehetővé teszi, hogy JavaScript kódot a szerveren futtass. A Node.js-t elsősorban a szerver oldali fejlesztéshez használják, például webes alkalmazások, API-k és más hálózati alkalmazások létrehozásához.

**Miért kell mindkettőt telepíteni?**

* **Különböző feladatok:** Az XAMPP-t elsősorban a dinamikus weboldalak fejlesztéséhez használják, míg a Node.js-t a szerver oldali logika megvalósításához.
* **Kiegészítik egymást:** A Node.js-szel létrehozott alkalmazások gyakran kommunikálnak egy adatbázissal, amelyet az XAMPP-ben található MySQL biztosíthat.

**Telepítési útmutató:**

1. **XAMPP telepítése:**
   * **Letöltés:** Látogass el az XAMPP hivatalos weboldalára ([https://www.apachefriends.org/index.html](https://www.google.com/url?sa=E&source=gmail&q=https://www.apachefriends.org/index.html)) és töltsd le a megfelelő verziót az operációs rendszeredhez.
   * **Telepítés:** Kövesd a telepítő varázsló utasításait. Általában csak a kívánt komponenseket kell kiválasztanod (Apache, MySQL, stb.).
   * **Indítás:** A telepítés után indítsd el az XAMPP Control Panel-t és indítsd el az Apache és a MySQL szolgáltatásokat.
2. **Node.js telepítése:**
   * **Letöltés:** Látogass el a Node.js hivatalos weboldalára ([https://nodejs.org/](https://www.google.com/url?sa=E&source=gmail&q=https://nodejs.org/)) és töltsd le a megfelelő LTS (Long Term Support) verziót.
   * **Telepítés:** Kövesd a telepítő varázsló utasításait. Általában csak a javasolt beállításokat kell elfogadnod.

**Ellenőrzés:**

* **XAMPP:** Nyisd meg a böngésződet és írd be a címsortba: http://localhost. Ha az XAMPP telepítése sikeres volt, akkor az Apache kezdőoldalát kell látnod.
* **Node.js:** Nyisd meg a terminált vagy a parancssort és írd be a következő parancsokat:
  + node -v: Ez kiírja a telepített Node.js verziót.
  + npm -v: Ez kiírja a telepített npm (Node Package Manager) verziót.

**Követelmény:**

1. Keressen és alkalmazzon egy **ingyenes reszponzív témát** az oldalaihoz. **(5 pont)**  
   A dokumentációban írja le, hogy melyik témát választotta.

Ötleteket a dokumentum végén talál.

1. **Főoldal menü**: Az első oldalon mutassa be a céget egy látványos weboldalon  **(5 pont)**
2. **Adatbázis menü:** a választott adatbázisból jelenít meg adatokat **(5 pont)**  
   Ehhez 3 tábla adatait használja fel az adatbázisból.
3. **Kapcsolat menü**: legyen egy kapcsolat űrlap, amelynek segítségével üzenetet **(5 pont)**

lehet küldeni az oldal tulajdonosa számára. Az elküldött Űrlap adatokat

mentse le az adatbázisba.

1. **Üzenetek menü**: Tegye lehetővé megtekinteni táblázatban az előző pontban **(5 pont)**

elküldött üzeneteket az adatbázisból fordított időrend szerint (a legfrissebb

legyen elől). Írja minden üzenethez a küldés idejét.

1. **CRUD menü**: Az oldalon valósítson meg egy CRUD (Create, Read, Update, Delete) **(15 pont)**

alkalmazást a választott adatbázis egyik táblájához.

CRUD funkciók: tábla megjelenítése; új rekord felvitele; adott rekord módosítása;

adott rekord törlése.

Mintát a dokumentum végén talál.

1. **OOP-Javascript menü**: Az oldalon készítsen egy kis alkalmazást, **(5 pont)**

amiben alkalmazza az objektum-orientált JavaScript elveit.

1. Alkalmazását töltse fel és valósítsa meg a megadott Linux-os tárhelyen.

(**Kötelező elem**! **A működés ez alapján lesz javítva**)

1. Használják a GitHub (github.com) verziókövető rendszert.   
   (**Kötelező elem**! **A forrás ez alapján lesz javítva**)

Ne csak a kész alkalmazást töltsék fel egy lépésben, hanem a részállapotokat is még legalább 5 lépésben személyenként.  
**A GIT-en saját nevet válasszanak, ami alapján be lehet azonosítani, hogy ki mit töltött fel.**

1. A **GitHub-on a projektmunka módszert alkalmazzák:** látszódjék, **(5 pont)**

hogy a csoport tagjai melyik részt készítették el.

1. Készítsen egy legalább 15 oldalas dokumentációt képernyőképekkel (**Kötelező elem**!),

amiben bemutatja alkalmazását és leírja, hogy az előző pontok feladatait hogyan valósította meg. **Fontos, hogy ez utóbbit leírja a dokumentációban, mert a feladat ez alapján lesz javítva.** A dokumentációban adja meg a GitHub projektjének URL címét is és a Linux és adatbázis belépési adatokat.

**1. Áttekintés és tervezés**

A feladat egy teljes körű webes alkalmazás fejlesztését kéri, amelynek során számos technológiát és koncepciót kell alkalmazni. A sikeres megvalósításhoz érdemes a következő lépéseket követni:

* **Technológiai stack kiválasztása:**
  + **Frontend:** React, Vue.js, Angular vagy más népszerű keretrendszer.
  + **Backend:** Node.js, PHP (Laravel) vagy más megfelelő technológia.
  + **Adatbázis:** MySQL, PostgreSQL vagy más relációs adatbázis.
  + **Verziókövető:** Git (GitHub)
* **Projektstruktúra tervezése:**
  + Komponensek, mappák és fájlok logikus elrendezése.
  + Adatáramlás megtervezése a frontend és backend között.
* **Felhasználói felület tervezése:**
  + Wireframe-ek készítése a különböző oldalakról.
  + Esztétikus és felhasználóbarát design kialakítása.

**2. Részletes lépések**

**Frontend fejlesztés:**

* **Téma kiválasztása:** Az ingyenes, reszponzív témák széles választéka elérhető a különböző keretrendszerekhez tartozó hivatalos vagy harmadik féltől származó könyvtárakban.
* **Komponensek létrehozása:** A feladatban szereplő menük, űrlapok, táblázatok és egyéb elemek elkészítése a választott keretrendszer segítségével.
* **Adatok megjelenítése:** Az adatbázisból származó adatok lekérdezése és megjelenítése a megfelelő komponensekben.
* **CRUD műveletek:** Az adatbázisban lévő adatok módosításának és törlésének lehetővé tétele.

**Backend fejlesztés:**

* **API létrehozása:** A frontend és a backend közötti kommunikáció biztosítása REST API-k segítségével.
* **Adatbázis kapcsolat:** Az adatbázishoz való csatlakozás és a szükséges lekérdezések végrehajtása.
* **Üzenetküldés:** Egy űrlap segítségével üzenetek küldése és tárolása az adatbázisban.
* **OOP JavaScript:** Az objektum-orientált programozás elveinek alkalmazása a JavaScript kódban.

**GitHub használat:**

* **Repository létrehozása:** A projekt számára egy új repository létrehozása a GitHub-on.
* **Commitok:** Rendszeres commitok készítése a változtatások nyomon követése érdekében.
* **Pull requestek:** A csapattagok által végzett módosítások egyeztetése és egyesítése.
* **Branches:** A fejlesztés különböző fázisainak elkülönítése ágak segítségével.

**Dokumentáció:**

* **Műszaki dokumentáció:** A projekt felépítésének, a használt technológiáknak és a megoldásoknak a részletes leírása.
* **Felhasználói dokumentáció:** A felhasználók számára készült útmutató az alkalmazás használatához.
* **Képernyőképek:** A fejlesztés különböző fázisaiban készült képernyőképek.

**3. További tippek**

* **Verziókezelés:** A Git használatával biztosítsd, hogy a csapattagok együttműködhessenek a projekten, és nyomon követhető legyen a fejlesztés története.
* **Tesztvezérelt fejlesztés:** Írj egységteszteket a kód minőségének biztosításához.
* **Responzivitás:** Győződj meg róla, hogy az alkalmazás megfelelően működik különböző képernyőméreteken.
* **Biztonság:** Védd meg az alkalmazást a biztonsági rések ellen.
* **Optimalizálás:** Tedd gyorsabbá és hatékonyabbá az alkalmazást.

**A dokumentációban szereplő minták és ötletek segítenek a feladat megoldásában. Ha bármilyen kérdésed van, ne habozz kérdezni!**

**Követelmények pontokba szedve: + Feladat kivitelezése:**

* 1. **Keressen és alkalmazzon egy ingyenes reszponzív témát az oldalaihoz. (5 pont)**

A reszponzív webdesign alapja a CSS (Cascading Style Sheets) nevű stíluslapnyelv. A CSS segítségével a weboldal elemei (szöveg, képek, táblázatok stb.) úgy vannak meghatározva, hogy a különböző képernyőméretekhez automatikusan igazodjanak.  
  
free responsive HTML CSS themes keresve:   
  
<https://themewagon.com/theme-tag/html5-css3/>

téma az utazás: <https://themewagon.github.io/Rhea/v1.0.0/>  
 **A dokumentációban írja le, hogy melyik témát választotta.**

**Ötleteket a dokumentum végén talál.**

* 1. **Főoldal menü: Az első oldalon mutassa be a céget egy látványos weboldalon (5 pont)  
       
     itt tartok**
  2. **Adatbázis menü: a választott adatbázisból jelenít meg adatokat (5 pont)  
     Ehhez 3 tábla adatait használja fel az adatbázisból.**
  3. **Kapcsolat menü: legyen egy kapcsolat űrlap, amelynek segítségével üzenetet (5 pont)**

**lehet küldeni az oldal tulajdonosa számára. Az elküldött Űrlap adatokat**

**mentse le az adatbázisba.**

* 1. **Üzenetek menü: Tegye lehetővé megtekinteni táblázatban az előző pontban (5 pont)**

**elküldött üzeneteket az adatbázisból fordított időrend szerint (a legfrissebb**

**legyen elől). Írja minden üzenethez a küldés idejét.**

* 1. **CRUD menü: Az oldalon valósítson meg egy CRUD (Create, Read, Update, Delete) (15 pont)**

**alkalmazást a választott adatbázis egyik táblájához.**

**CRUD funkciók: tábla megjelenítése; új rekord felvitele; adott rekord módosítása;**

**adott rekord törlése.**

**Mintát a dokumentum végén talál.**

* 1. **OOP-Javascript menü: Az oldalon készítsen egy kis alkalmazást, (5 pont)**

**amiben alkalmazza az objektum-orientált JavaScript elveit.**

* 1. **Alkalmazását töltse fel és valósítsa meg a megadott Linux-os tárhelyen.**

**(Kötelező elem! A működés ez alapján lesz javítva)**

* 1. **Használják a GitHub (github.com) verziókövető rendszert.   
     (Kötelező elem! A forrás ez alapján lesz javítva)**

**Ne csak a kész alkalmazást töltsék fel egy lépésben, hanem a részállapotokat is még legalább 5 lépésben személyenként.  
A GIT-en saját nevet válasszanak, ami alapján be lehet azonosítani, hogy ki mit töltött fel.**

* 1. **A GitHub-on a projektmunka módszert alkalmazzák: látszódjék, (5 pont)**

**hogy a csoport tagjai melyik részt készítették el.**

* 1. **Készítsen egy legalább 15 oldalas dokumentációt képernyőképekkel (Kötelező elem!),**

**amiben bemutatja alkalmazását és leírja, hogy az előző pontok feladatait hogyan valósította meg. Fontos, hogy ez utóbbit leírja a dokumentációban, mert a feladat ez alapján lesz javítva. A dokumentációban adja meg a GitHub projektjének URL címét is és a Linux és adatbázis belépési adatokat.**