Komplexní Dokumentace: Plánovač Rozvrhu (React Aplikace) - Finální Verze

1. Úvod

1.1. Cíl aplikace

Cílem této webové aplikace je poskytnout studentům Západočeské univerzity v Plzni (ZČU) nástroj pro efektivní plánování a vizualizaci jejich studijního rozvrhu. Aplikace umožní studentům sestavit si rozvrh ještě před oficiálním zápisem do univerzitního systému IS/STAG, čímž jim pomůže lépe plánovat a optimalizovat svůj čas. Studenti si budou moci načíst předměty a jejich rozvrhové akce, zvolit si preference (např. volný den) a nechat si automaticky vygenerovat možné varianty rozvrhu. Aplikace klade důraz na přehledné uživatelské rozhraní a smysluplnou funkcionalitu.

1.2. Cíloví uživatelé

Aplikace je primárně určena pro studenty Západočeské univerzity v Plzni, kteří si mohou svůj rozvrh sestavovat sami.

2. Fungování rozvrhů, předmětů a rozvrhových akcí na univerzitě

2.1. Rozvrh (Schedule)

Rozvrh je vizuální reprezentací zapsaných předmětů a jejich rozvrhových akcí v průběhu týdne. Zobrazuje se ve stylu českých rozvrhů:

- **Struktura:** Tabulka, kde levý první sloupec obsahuje dny v týdnu (pondělí až neděle). Každý řádek v tabulce představuje jeden den.
- **Časové bloky:** Horní řádek (záhlaví) obsahuje jednotlivé časové bloky. Konkrétně se jedná o 14 bloků v časovém rozmezí od 7:30 do 20:10, s desetiminutovými přestávkami mezi nimi. Tyto bloky jsou definovány pouze v záhlaví; pozice rozvrhových akcí v řádcích dnů se vypočítává relativně k těmto časovým blokům. Například přednáška od 7:30 do 8:30 může přesahovat první časový blok (např. 7:30-8:15) a zasahovat do druhého (např. 8:25-9:10).

2.2. Předmět (Course)

Předmět reprezentuje vyučovanou látku na univerzitě.

- **Identifikace:** Každý předmět spadá pod nějaké pracoviště (např. katedru) a má unikátní kód. Graficky se označuje zkratkou katedry a předmětu (např. KMA/MA2).
- **Kredity:** Každý předmět má přiřazené kreditové ohodnocení.
- **Rozvrhové akce:** Každý předmět má definovaný plán vyučovacích hodin/událostí, tzv. rozvrhové akce.

- **Podmínky zápisu:** Pro každý předmět je stanoven maximální (nebo doporučený/povinný) počet rozvrhových akcí určitého typu (přednáška, cvičení, seminář), které si student musí nebo může zapsat pro řádný zápis. Například předmět KIV/PPA může vyžadovat zápis dvou rozvrhových akcí typu přednáška a jedné rozvrhové akce typu cvičení.

2.3. Rozvrhová akce (Course Event)

Rozvrhová akce je konkrétní událost/akce v rámci předmětu, jako je přednáška, cvičení nebo seminář.

- **Základní atributy:**
    - **Čas:** Začátek a konec akce (časově).
    - **Den:** Specifický den v týdnu. V datovém modelu je den reprezentován číselně (0 = pondělí, 1 = úterý, ..., 6 = neděle), což je potřeba ošetřit pro zobrazení.
    - **Opakování:** Určuje, zda se akce koná každý týden, každý sudý týden, nebo každý lichý týden.
    - **Platnost:** Časové období (datum od-do), po které je rozvrhová akce platná. Může se změnit např. při nedostupnosti místnosti.
    - **Místnost:** Učebna, kde se akce koná.
    - **Kapacita:** Maximální počet studentů a aktuálně zapsaný počet studentů.
    - **Vyučující:** Osoba zodpovědná za akci (přednášející, cvičící).
    - **Typ:** Přednáška, cvičení, seminář. Existují i speciální typy jako "zápočet" a "zkouška". Typ akce může být vizuálně odlišen barvou (např. červená pro přednášky, zelená pro cvičení, žlutá pro semináře).
    - **Předmět:** Ke kterému kurzu akce patří.
    - **Poznámka (Dodatek):** Doplňující text nebo popis.
    - **Rok a Semestr:** Akademický rok a semestr, ke kterému akce náleží.
- **Překrývající se rozvrhové akce:**
    - Pokud se dvě nebo více akcí časově překrývají (např. Přednáška KIV/PPA 9:30-11:00 a Cvičení KMA/MA2 10:00-12:00), je nutné je graficky přehledně prezentovat.
    - V rozvrhu se takové překrytí projeví vytvořením dalšího grafického řádku (úrovně) pro daný den. Akce začínající později se zobrazí na tomto novém,

nižším řádku/úrovni. Akce začínající dříve zůstává na původní úrovni (úroveň 0).

- **Virtuální rozvrhová akce:**
    - Jedná se o placeholder, dočasnou akci, často používanou před začátkem předzápisů nebo při úpravách plánu akcí předmětu.
    - Studenti jsou do ní dočasně zařazeni a po vytvoření řádných akcí se z virtuální odhlásí a přihlásí na reálnou.
    - Budou se zobrazovat pod hlavní tabulkou rozvrhu. Zobrazuje název předmětu, kapacitu, semestr a dodatek (často s informací o dočasnosti). Tyto akce nebude možné zapsat ani odepsat prostřednictvím uživatelského rozhraní aplikace.
    - Pozná se také tím, že nemá přiřazenou místnost (je isVirtual).
- **Propojené rozvrhové akce (Rozvrhové skupiny):**
    - Některé akce mohou být propojeny tak, že zápis jedné automaticky znamená zápis druhé (např. pomocí groupId). Například předmět UJP/AEP má cvičení ve dvojicích; zápis cvičení v úterý automaticky zapíše i propojené cvičení v pátek.
- **Rozvrhové kroužky:**
    - Označují uskupení studentů. Rozvrhová akce může být přiřazena k těmto kroužkům, čímž se omezí pouze pro studenty v daných kroužcích. groupId v CourseEventClass může toto reprezentovat.
- **Vizuální konzistence:** Všechny komponenty reprezentující rozvrhové akce (např. ScheduleEvent.jsx) by měly mít stejnou výšku pro konzistentní vzhled v rámci jednoho řádku/úrovně rozvrhu.

2.4. Zápis (Enrollment)

- Studenti si volí předměty a k nim příslušné rozvrhové akce.
- **Grafický předzápis:** Univerzitní systém typicky poskytuje rozhraní (grafický předzápis), kde si studenti mohou akce vybírat. Kliknutím na akci v rozvrhu se zobrazí více informací a checkbox pro označení volby k zápisu.
- Odškrtnutí akce může zobrazit jiné dostupné akce stejného typu (např. jiné cvičení).
- Samotný zápis se neprovádí ihned zaškrtnutím, ale až po potvrzení (např. tlačítkem "Uložit změny").

- Zápis je podmíněn splněním určitých podmínek, jako je volná kapacita akce.

3. Portál univerzity IS/STAG (University Portal IS/STAG)

3.1. Obecný popis

IS/STAG je univerzitní informační systém Západočeské univerzity v Plzni. Slouží mimo jiné k evidenci studentů, předmětů, rozvrhových akcí a umožňuje studentům provádět zápis předmětů. Aplikace bude využívat jeho veřejné API endpointy pro načítání dat.

3.2. Relevantní objekty v IS/STAG

Pro potřeby aplikace jsou relevantní následující datové objekty v IS/STAG:

- Rozvrhová akce
- Předmět
- Studijní plán/program
- Studijní obor

3.3. Uživatelská identita a role v IS/STAG

- **ORION konto:** Každá fyzická osoba (student, zaměstnanec) má na ZČU tzv. ORION konto (např. sbriza), které slouží k přihlášení do STAGu a identifikaci na univerzitě.
- **IS/STAG identity (role):** K jednomu ORION kontu může být přiřazeno více identit, kde každá identita reprezentuje specifickou roli uživatele v systému. Například student může mít identitu A24B0093P, která ho reprezentuje v roli studenta pro jeho studijní program. Pokud by tatáž osoba studovala další obor na jiné fakultě nebo začala vyučovat, získala by další identity pro tyto role.
- **IS/STAG uživatelské jméno:** Každá jednotlivá role (identita) má svůj jedinečný identifikátor napříč celým systémem, nazvaný "IS/STAG uživatelské jméno". Tento identifikátor se používá v parametru stagUser při volání API služeb vyžadujících specifickou roli. Může se lišit od přihlašovacího jména ORION konta.
- **Konfigurace přihlašování:** IS/STAG může být nakonfigurován pro přihlašování ke každé roli zvlášť (pomocí "IS/STAG uživatelského jména" a hesla z databáze STAGu) nebo pro přihlašování osoby přes externí systém (např. Shibboleth, LDAP), kde se uživatel přihlásí svým ORION kontem a následně může vystupovat pod všemi svými rolemi specifikováním parametru stagUser.

4. API Univerzity (STAG API)

4.1. Přístup a autentizace

- **Protokol:** Veškeré webové služby STAG API jsou publikovány výhradně prostřednictvím protokolu HTTPS.
- **Anonymní vs. autentizované volání:** Některé služby lze volat anonymně, zatímco mnohé vyžadují přihlášení. Některé služby mohou vracet více informací, pokud jsou volány přihlášeným uživatelem s určitou rolí.
- **Autentizační mechanismus:** Pro přenos přihlašovacích údajů se používá standardní HTTP mechanismus Basic Authentication.
    - Lze předat kombinaci uživatelského jména a hesla (méně časté a nedoporučené pro klientské aplikace).
    - Lze předat tzv. **uživatelský ticket** (náhodně generovaný řetězec serverem) jako uživatelské jméno v HTTP Basic, přičemž heslo je prázdný řetězec.
- **Získání uživatelského ticketu (**stagUserTicket**):**

1. **WSCOOKIE:** Po úspěšném zavolání webové služby, která akceptovala přihlášení, server vrací cookie s názvem WSCOOKIE, jejíž hodnotou je uživatelský ticket. Tuto cookie lze následně použít pro další volání.

2. **Přesměrování na přihlašovací stránku STAGu:** Aplikace může přesměrovat uživatele na speciální adresu modulu webových služeb STAGu (např. https://stag-ws.zcu.cz/ws/login) s parametrem originalURL. Tento originalURL specifikuje adresu, na kterou bude uživatel přesměrován zpět po úspěšném přihlášení.
    - Doporučuje se formulářové ověření: https://stag-demo.zcu.cz/ws/login?originalURL={URL_APLIKACE}.
    - Lze použít parametr onlyMainLoginMethod=1 pro použití pouze hlavní přihlašovací metody školy (např. Shibboleth).
    - Pro ticket s delší platností (až 90 dní) lze přidat parametr &longTicket=1.
    - Adresa v originalURL musí být URL enkódovaná.
    - Po úspěšném přihlášení server přesměruje uživatele zpět na originalURL a přidá query parametry:
        - stagUserTicket: Uživatelský ticket. Pokud se uživatel přihlásí jako anonymní, hodnota je "anonymous".
        - stagUserInfo: Base64 enkódovaný JSON obsahující informace o přihlášeném uživateli, především seznam jeho rolí v IS/STAG. Pokud anonymní, JSON obsahuje prázdný seznam rolí.

- **Použití ticketu:** Aplikace následně používá získaný stagUserTicket pro volání dalších služeb WS IS/STAG, typicky v HTTP Basic hlavičce (ticket jako username, prázdné heslo).
- **Neukládání tokenu:** API token se v aplikaci nikde trvale neukládá; používá se pouze pro aktuální session načítání dat.
- localhost **pro debug:** Je třeba ošetřit zabezpečené přihlášení do STAGu i při vývoji na localhost. To obvykle znamená, že originalURL musí být adresa dostupná z internetu, nebo je nutné použít specifické postupy pro vývojové prostředí, pokud je STAG podporuje.
- **Parametr** stagUser**:** U služeb, kde záleží na roli volajícího, je nutné vyplnit parametr stagUser s hodnotou "IS/STAG uživatelského jména" dané role.
- **Chybové kódy:** V případě neúspěšného přihlášení vrací server HTTP kódy 401 nebo 403 [is-stag.zcu.cz_napoveda_web-services_ws_prihlasovani.html.pdf, Page 2, "V případě, že webová služba vyžaduje přihlášení, které však není z jakéhokoliv důvodu splněno, vrací server HTTP chybové kódy 401 či 403."].

4.2. Načítání dat

- **Období:** Načítání dat (předmětů, rozvrhových akcí) musí probíhat pro uživatelem stanovené období (akademický rok a semestr).
- **Způsoby načtení předmětů:**
    1. **Podle kódu předmětu:** Uživatel zadá kódy předmětů (např. KMA/MA2), maximálně 20 najednou.
    2. **Ze studijního plánu studenta:** Vyžaduje načtení identit uživatele, volbu identity a následné použití ID identity pro volání API endpointu.
- **Shrnující dialog:** Import/načtení předmětů by mělo být zakončeno dialogem, který shrne, jaké předměty byly úspěšně přidány (s počtem rozvrhových akcí) a jaké se nepodařilo načíst (např. neexistují).

4.3. Relevantní STAG API Endpoints

Následující typy endpointů budou pravděpodobně potřeba:

- **Pro informace o uživateli a jeho rolích:**
    - help/getStagUserListForActualUser nebo help/getStagUserListForActualUserV2: Vrátí seznam IS/STAG rolí aktuálně přihlášeného uživatele.

    - help/getStagUserListForLoginTicket nebo help/getStagUserListForLoginTicketV2: Vrátí seznam IS/STAG rolí pro uživatele identifikovaného ticketem.
    - users/getStagUserListForExternalLogin nebo users/getStagUserListForExternalLoginV2: Vrátí seznam IS/STAG rolí na základě externího loginu.
- **Pro načítání předmětů:**
    - predmety/najdiPredmety: Vyhledá předměty podle kritérií jako název, pracoviště (katedra), zkratka, rok. Parametry: nazev, pracoviste, zkratka, rok, lang.
    - predmety/getPredmetyByKatedra: Vrátí seznam předmětů katedry. Parametry: katedra, rok, semestr, jenNabizeneECTSPrijezdy, lang.
    - predmety/getPredmetyByStudent: Vrátí seznam předmětů studenta zapsaných v IS/STAG (může být použito pro ověření nebo pro načítání dle studijního plánu). Parametry: osCislo, semestr, rok, nevracetUznane, lang.
    - predmety/getPredmetInfo: Vrátí kompletní informace o jednom předmětu. Parametry: katedra, zkratka, rok, lang.
- **Pro načítání rozvrhových akcí:**
    - rozvrhy/getRozvrhoveAkce: Vrátí seznam rozvrhových akcí podle zadaných kritérií (povinně rok, semestr a alespoň jedno další kritérium). Parametry: pracoviste, zkrPredm, zkrBudovy, cisloMistnosti, ucitIdno, den, rokVarianty, semestr, typ, owner, platnost, lang.
- **Pro informace o akademickém kalendáři:**
    - kalendar/getAktualniObdobiInfo: Vrátí informace o aktuálním akademickém roce, semestru atd.
    - kalendar/getHarmonogramRoku: Vrátí harmonogram zadaného akademického roku. Parametr: rok.
- **Pro číselníky (pokud je potřeba):**
    - ciselniky/getCiselnik: Vrátí obsah číselníku zadaného doménou (např. typy akcí, typy opakování, dny v týdnu, pokud nejsou přímo v datech rozvrhových akcí). Parametry: domena, lang.
    - ciselniky/getSeznamDomen: Vrátí seznam názvů všech domén číselníků.
- **Pro informace o studentech (pokud je nutné načítat identity přímo):**

- student/getStudentInfo: Vrátí informace o studentovi. Parametry: osCislo, rok, zobrazovatSimsUdaje, lang.
- users/getOsobniCislaByExternalLogin: Vrátí osobní čísla studenta dle externího loginu. Parametry: login, pouzeStudujici.

4.4. Formát dat

API služby typicky podporují výstupní formáty XML a JSON, případně YAML a CSV pro některé endpointy. Pro webovou aplikaci bude preferován formát JSON.

5. Struktura a Funkcionality Aplikace

Celá aplikace bude sledovat veškerá data pomocí instance WorkspaceService, která bude mít uložené aktuálně načtené kurzy/předměty, aktuálně zapsané rozvrhové akce, aktuální podobu rozvrhu a aktuální stav preferencí/vlastností ke generování rozvrhu.

5.1. Hlavní stránky/okna

Aplikace bude mít následující hlavní části/stránky:

1. **Editor rozvrhu (SchedulePage / editorPage.jsx):** Hlavní pracovní plocha pro vytváření a úpravu rozvrhu.
2. **FAQ (FAQPage):** Stránka s často kladenými otázkami.
3. **Úvodní stránka (LandingPage):** Vstupní stránka aplikace. Tyto tři smysluplné obrazovky splňují základní požadavek na počet různých obrazovek.

5.2. Hlavní komponenty React

Struktura projektu je organizována následovně (hlavní komponenty editoru):

- Sbriza/src/editor/editorPage.jsx: Komponenta reprezentující celou stránku editoru a její layout. Obsahuje horní lištu (AppBar) a hlavní layout s levým, středovým a pravým oddílem.
    - **Topbar (Horní lišta):** Na stránce editoru bude obsahovat tlačítka k načtení/uložení předmětů, načtení/uložení stavu pracovní plochy/projektu, tlačítko na vytisknutí rozvrhu a tlačítko na stažení rozvrhu jako .png obrázku.
    - Levý a pravý sidebar jsou měnitelné šířky, což přispívá k responzivitě aplikace.
    - **Responzivní chování na mobilních zařízeních (malé displeje):**
        - Obsah Topbaru se zkonsoliduje do menu (např. "hamburger" ikona).

        - Levý i pravý sidebar se transformují na SwipeableDrawer MUI komponenty.
        - Uvnitř SwipeableDrawer (nebo jednoho společného pro oba, pokud designově vhodnější) bude MUI Tab komponenta umožňující přepínání mezi obsahem původního levého sidebaru (strom předmětů) a pravého sidebaru (seznam preferencí).

- Sbriza/src/editor/courses.jsx (**Levý sidebar** - CourseBar): Zobrazuje načtené předměty a jejich rozvrhové akce ve stromové struktuře KATEDRA/PŘEDMĚT/Rozvrhová akce. Katedra a Předmět uzly slouží primárně pro rozkliknutí a zobrazení podřazených položek. Používá komponentu RichTreeView z MUI. Přijímá courses (z workspace.courses) a workspace jako props. Komponenta se musí korektně aktualizovat při změnách v WorkspaceService.courses (např. odstranění předmětu). V tomto sidebaru bude implementována funkce vyhledávání, která uživateli umožní filtrovat a rychle nalézt specifický předmět mezi načtenými.
    - Uzel reprezentující **Předmět** ve stromu:
        - Musí mít tlačítko na odstranění tohoto předmětu. Odstranění proběhne tak, že se nejprve zruší zápis všech rozvrhových akcí daného předmětu v aktuálním rozvrhu studenta (workspace.schedule) a poté se předmět odstraní ze seznamu workspace.courses.
        - Pod názvem předmětu se zobrazí jeho kreditní ohodnocení.
    - Sbriza/src/editor/components/courseItem.jsx: Komponenta (TreeItem) reprezentující **Rozvrhovou akci** ve stromu CourseBar.
        - Zobrazuje: typ akce (např. přednáška), den v týdnu, čas začátku a konce, místnost, maximální kapacitu, aktuálně zaplněnou kapacitu (např. 5/10), typ opakování akce (např. sudý týden) a případné poznámky k akci.
        - Obsahuje přepínač (toggleButton) pro zápis/odzápis dané rozvrhové akce do/z aktuálního rozvrhu studenta (workspace.schedule).
- Sbriza/components/ScheduleBox.jsx (nebo Sbriza/src/editor/schedule.jsx) (**Středový obsah**): Komponenta zobrazující samotný rozvrh formou tabulky. Formát rozvrhu je takový, že sloupce reprezentují časové bloky a řádky dny v týdnu.

- Pokud je rozvrh příliš široký a přesahuje viditelnou oblast, mělo by být možné v něm horizontálně skrolovat. Je nutné zajistit správné scrollování obsahu rozvrhu, přičemž záhlaví s časovými bloky by mělo zůstat viditelné nebo se scrollovat synchronizovaně s obsahem (vyhnout se nezávislému scrollování pouze záhlaví nebo rozbitému scrollování v overflow oblasti).
- Záhlaví s časovými bloky by mělo mít číselné označení bloků od 1 do 14.
- Pro optimalizaci a zjednodušení není nutné explicitně vykreslovat svislé čáry oddělující jednotlivé časové bloky v těle rozvrhu.
- Přijímá events z workspace.schedule.events jako props.
- **Dynamické zobrazení a interakce:** Rozvrhové akce lze zapisovat/odpisovat i přímo v této tabulce. Chování zobrazení akcí pro daný předmět je následující:
    - Pokud je zapsán správný počet rozvrhových akcí pro daný předmět (dle CourseClass.maxEnrollments), v rozvrhu se zobrazí pouze tyto zapsané akce s možností jejich odzápisu.
    - Pokud není zapsán dostatečný počet rozvrhových akcí pro řádné uznání zápisu předmětu, zobrazí se v rozvrhu všechny dostupné rozvrhové akce tohoto předmětu (zejména ty typy, které chybí), aby si uživatel mohl graficky prohlédnout, kam by se mu daná akce hodila zapsat, a mohl ji zde přímo zapsat.
- Sbriza/components/ScheduleEvent.jsx: Komponenta reprezentující graficky jednu rozvrhovou akci v ScheduleBox.
    - Zobrazuje: název předmětu (formát KATEDRA/PŘEDMĚT např. KIV/PPA), obsazenost/kapacitu (např. 17/20), vyučujícího, místnost, opakovatelnost (např. každý sudý týden).
    - Typ rozvrhové akce je znázorněn barevným odstínem pozadí komponenty (např. červená = přednášky, zelená = cvičení, žlutá = seminář).
    - Šířka komponenty odpovídá délce trvání rozvrhové akce (funguje jako časová osa v řádku dne). Musí být adaptivní na různé šířky.
    - Měla by být kompaktní a mít konzistentní výšku s ostatními ScheduleEvent.jsx komponentami v daném řádku/úrovni.
    - Umožňuje přímý zápis/odzápis akce (např. kliknutím, checkboxem).

    - ScheduleEventDropdown.jsx: Komponenta, která se zobrazí po kliknutí na ScheduleEvent v ScheduleBox a ukáže další informace o akci.

- Sbriza/src/editor/components/PropertiesBar.jsx (**Pravý sidebar** - PreferenceBar): Zobrazuje seznam aktuálně nastavených preferencí/parametrů pro generování rozvrhu, načtených z WorkspaceService.priorities. Položky jsou řazeny od nejdůležitější (priorita 1 nahoře) po nejméně důležitou (nejmenší priorita, max 99). Sidebar se musí korektně aktualizovat a zobrazovat aktuální stav WorkspaceService.priorities. Tlačítko pro spuštění generování rozvrhu bude umístěno v tomto pravém sidebaru.
    - Sbriza/src/editor/components/PropertyItem.jsx: Komponenta reprezentující jednu položku (preferenci) v PropertiesBar.
        - Umožňuje měnit prioritu parametru pomocí šipek nebo přímým zápisem čísla priority.
        - Zajišťuje, že priority tvoří aritmetickou posloupnost s krokem 1 (chová se jako index pole); např. po prioritě 5 musí následovat 6, ne 11.
        - Obsahuje tlačítko pro odstranění preference.
        - Obsahuje přepínač (toggleButton) pro aktivaci/deaktivaci (parametr skip v datech preference) preference během generování.
    - Sbriza/src/editor/components/PropertyCreateDialog.jsx: Popup okno pro vytváření nových preferencí. Musí podporovat různé typy preferencí, které mohou vyžadovat odlišné vstupní parametry (např. "volný den" vs. "preferuj vyučujícího pro konkrétní typ akce konkrétního předmětu").

5.3. Třídy

Aplikace bude využívat následující třídy pro organizaci dat:

- CourseClass (Reprezentuje předmět)
    - **Vlastnosti:**
        - name (string): Název předmětu.
        - departmentCode (string): Zkratka katedry.
        - courseCode (string): Zkratka předmětu.
        - credits (number): Kreditové ohodnocení.
        - maxEnrollments (object): Požadovaný/maximální počet zapsaných akcí daného typu, např. { lecture: 2, practical: 1, seminar: 0 }.

        - events (array of CourseEventClass): Rozvrhové akce tohoto předmětu.
        - stagId (string/number): ID předmětu ze STAG API.
        - semester (string): Semestr (např. "LS", "ZS").
        - selected (boolean): Indikuje, zda je předmět vybrán uživatelem (např. pro načtení akcí).
    - **Metody:**
        - constructor({name, departmentCode, courseCode, credits, maxEnrollments, events, stagId, semester}).
        - areConditionsMet(selectedEvents): Vrací true, pokud je pro daný výběr selectedEvents splněn požadavek na maxEnrollments (např. student si nezapsal více cvičení, než je povoleno, nebo naopak má zapsaný požadovaný počet).
        - getCourseEvents(): Vrací všechny rozvrhové akce předmětu (this.events) [Struktura react.pdf, "getCourseEvents() {"].
- CourseEventClass (Reprezentuje rozvrhovou akci)
    - **Vlastnosti:**
        - id (string/number): Unikátní identifikátor akce.
        - startTime (string): Čas začátku (např. "10:00").
        - endTime (string): Čas konce (např. "11:30").
        - day (number): Den v týdnu (0 pro pondělí, 1 pro úterý, ..., 6 pro neděli).
        - recurrence (string): Opakování (např. "KAŽDÝ TÝDEN", "SUDÝ TÝDEN", "LICHÝ TÝDEN").
        - validityStart (string/Date): Datum začátku platnosti akce.
        - validityEnd (string/Date): Datum konce platnosti akce.
        - course (string/CourseClass): Reference na předmět, ke kterému akce patří.
        - room (string): Místnost.
        - type (string): Typ akce (např. "PŘEDNÁŠKA", "CVIČENÍ", "SEMINÁŘ").

        - instructor (string): Jméno vyučujícího.
        - currentCapacity (number): Aktuální počet zapsaných studentů.
        - maxCapacity (number): Maximální kapacita.
        - note (string): Doplňující poznámka.
        - isVirtual (boolean): true, pokud se jedná o virtuální akci (pozná se např. absencí místnosti).
        - year (number/string): Akademický rok.
        - semester (string): Semestr.
        - groupId (string/number, optional): ID rozvrhové skupiny (kroužku).
    - **Metody:**
        - constructor(data).
        - Může obsahovat getter metodu pro převod číselného dne na textový řetězec (např. getDayAsString()).
- ScheduleClass (Reprezentuje sestavený rozvrh studenta)
    - **Vlastnosti:**
        - events (array): Seznam instancí CourseEventClass, které si student zapsal do svého rozvrhu.
        - Grafická struktura pro zobrazení: Rozvrh je pole dnů; každý den je pole úrovní; každá úroveň je pole rozvrhových akcí. Rozvrh = [Pondělí: [úroveň0: [rozvrhové akce], úroveň1: [překrývající se rozvrhové akce]]], atd.. Seznam rozvrhových akcí je seřazen dle času začátku. Překrývající se rozvrhové akce se graficky zobrazí pod sebou na nové úrovni/řádku, přičemž akce začínající později se umisťuje na spodní úroveň.
    - **Metody:**
        - constructor().
        - addEvent(event): Přidá jednu CourseEventClass do rozvrhu.
        - addEvents(eventList): Přidá pole instancí CourseEventClass do rozvrhu.
        - popEvent(): Odebere poslední přidanou akci z rozvrhu. (Poznámka: Bude pravděpodobně potřeba sofistikovanější metoda pro odebrání konkrétní akce.)

- WorkspaceService (Spravuje celkový stav pracovní plochy editoru)
    - **Vlastnosti:**
        - semester (string): Aktuálně zvolený semestr (např. "LS", "ZS").
        - year (string/number): Aktuálně zvolený akademický rok.
        - courses (array of CourseClass): Seznam všech načtených předmětů.
        - schedule (ScheduleClass instance): Aktuálně sestavený rozvrh studenta.
        - preferences (object): Slovník preferencí studenta pro generování rozvrhu, kde klíč je priorita (číslo) a hodnota je objekt s detaily preference. Např.: { 1: { type: 0, day: 2, skip: false }, 2: { type: "PREFER_INSTRUCTOR_FOR_SUBJECT_EVENT_TYPE", subjectCode: "KIV/PPA1", eventType: "CVIČENÍ", instructorName: "Dr. Novák", skip: true } }.
    - **Metody:**
        - constructor().
        - saveWorkspace(): Uloží aktuální stav WorkspaceService (semester, year, courses, schedule, preferences) do localStorage.
        - loadWorkspace(): Načte stav WorkspaceService z localStorage.
        - saveScheduleImage(): Vygeneruje obrázek aktuálního rozvrhu (např. .png), potenciálně pomocí externí knihovny jako html2canvas.
        - generateSchedule(): Hlavní metoda pro algoritmus generování rozvrhu na základě preferencí. (Detail viz sekce 5.4.4).
        - clearWorkspace(): Resetuje pracovní plochu do výchozího prázdného stavu (vynuluje semester, year, courses, schedule, preferences).
        - Může obsahovat další metody pro manipulaci s courses a schedule (např. přidání/odebrání předmětu, přidání/odebrání akce do/z rozvrhu, které by se volaly z komponent).

5.4. Funkcionality

5.4.1. Načítání předmětů

Uživatel si bude moci načíst předměty několika způsoby:

- **Ze STAG API:**
    - Podle kódu předmětu (např. KIV/PPA), maximálně 20 předmětů najednou.
    - Ze studijního plánu studenta (vyžaduje přihlášení a výběr identity).
    - Načítání probíhá pro zvolený akademický rok a semestr.
- **Z JSON souboru:** Možnost importovat předměty ze souboru.
- **Z kódu/textu:** Možnost zkopírovat data předmětů (např. od kamaráda) a načíst je.
- Po každém načtení se zobrazí dialog se shrnutím úspěšně načtených předmětů (včetně počtu jejich rozvrhových akcí) a neúspěšných (např. neexistující).

5.4.2. Manuální tvorba předmětů a akcí

Pro offline použití, testování, nebo pokud STAG API není dostupné:

- Uživatel může manuálně vytvořit vlastní předměty (např. KMA/MA2).
- Do manuálně vytvořených předmětů může přidávat jednotlivé rozvrhové akce.
- Pro účely ladění bude možné generovat náhodné rozvrhové akce k předmětům.

5.4.3. Správa pracovní plochy (Workspace)

Uživatel si bude moci uložit a načíst aktuální stav pracovní plochy ("editoru"):

- Zahrnuje: všechny načtené předměty, stav zapsaných rozvrhových akcí (aktuální rozvrh) a nastavení preferencí pro generování.
- Ukládání/načítání do/z:
    - **URL adresy:** Pro zachování stavu při přesměrování (např. během STAG autentizace).
    - **Úložiště prohlížeče (**localStorage**):** Pro persistentní uložení na straně klienta.
    - **JSON souboru:** Pro export/import pracovní plochy.
- Možnost vymazat/resetovat pracovní plochu (clearWorkspace()). Tato funkcionalita odpovídá požadavku na ukládání a načítání dat z disku a ukládání stavu pro pokračování v práci.

5.4.4. Generování rozvrhu (generateSchedule())

Algoritmus se pokusí sestavit rozvrh dle zvolených předmětů, jejich podmínek a uživatelských preferencí.

- **Vstup:** Seznam předmětů k zapsání, uživatelské preference s prioritami.

- **Logika:**
    - Zpracovává preference postupně od nejvyšší priority k nejnižší.
    - Pro každou preferenci se pokusí vybrat rozvrhové akce tak, aby byla preference splněna a zároveň byly splněny podmínky pro zápis předmětů (např. povinný počet přednášek/cvičení pro každý předmět).
    - **Nedestruktivní operace:** Během generování se neupravují originální data (seznamy akcí, stav zapsaných akcí). Zápis/odzápis se provede až na konci po úspěšném vygenerování celého rozvrhu.
    - **Bez překryvů:** Vygenerované rozvrhové akce pro studenta se nesmí časově překrývat.
    - **Kontrola platnosti:** Po aplikaci každé preference se ověří, zda je stále možné sestavit platný rozvrh (tj. zda lze zapsat požadovaný počet akcí pro všechny vybrané předměty).
    - **Handling nesplnitelných preferencí:**
        - Pokud není možné vyhovět všem preferencím, vyhledávání se standardně zastaví.
        - Zobrazí se dosud vygenerovaný (částečný) rozvrh (zapíše se do instance WorkspaceService).
        - Aplikace vyznačí, které preference byly úspěšně aplikovány a které nikoliv.
    - **Přeskakování preferencí:** Preference s nastavenou hodnotou skip: true (nebo podobným příznakem) se ignoruje a pokračuje se na další, pokud existuje.
- **Typy preferencí (zatím definované):**

1. **Volný den:** Uživatel si zvolí konkrétní den, který chce mít volný (např. středa). (Struktura dat např. { type: 0, day: 0-6, skip: boolean } kde day:0 je pondělí).

2. **Volný časový blok:** Uživatel si zvolí časové období v konkrétním dni, které chce mít volné (např. Po 10:00-12:00 kvůli obědu). (Struktura dat např. { type: 1, day: 0-6, timeStart: "HH:MM", timeEnd: "HH:MM", skip: boolean }).

3. **Preferuj vyučujícího pro typ akce předmětu:** Uživatel chce konkrétního vyučujícího pro specifický typ rozvrhové akce (např. cvičení) konkrétního předmětu (např. KIV/PPA1). Struktura dat by mohla být: { type: "PREFER_INSTRUCTOR_FOR_SUBJECT_EVENT_TYPE", subjectCode: "KIV/PPA1", eventType: "CVIČENÍ", instructorName: "Jméno Příjmení", skip: boolean }.

- **Správa priorit preferencí:**
    - Každá preference má prioritu (celé číslo od 1 do 99, kde 1 je nejvyšší).
    - Pořadí v PropertiesBar je od nejvyšší priority dolů.
    - Prioritu lze měnit (šipky, přímý vstup).
    - Systém zajišťuje, že priority jsou vždy souvislou řadou (jako indexy pole, např. 1, 2, 3, ...).

5.4.5. Vizualizace rozvrhu

- **Styl:** Český styl rozvrhu – dny v týdnu v prvním sloupci zleva, časové bloky v záhlaví nahoře.
- **Časové bloky:** Od 7:30 do 20:10, celkem 14 bloků s 10minutovými přestávkami. Tyto bloky v záhlaví budou mít číselné označení 1 až 14. Akce se vkládají dynamicky dle jejich časů.
- **Zobrazení překryvů:** Překrývající se akce se zobrazují pod sebou ve více "podřádcích" (úrovních) pro daný den, přičemž akce začínající později se zobrazí na nižší úrovni.

5.4.6. Interakce s rozvrhem

- **Výběr/zápis akcí:** Uživatel si vybírá rozvrhové akce k zapsání. Toto je možné provést:
    - Pomocí toggleButtonů v CourseItem.jsx v levém sidebaru (CourseBar).
    - Přímo v tabulce rozvrhu (ScheduleBox) kliknutím/interakcí s komponentou ScheduleEvent.jsx.
- **Dynamické zobrazení v** ScheduleBox **dle stavu zápisu:**
    - Pokud jsou pro daný předmět splněny podmínky zápisu (např. zapsány 2 přednášky a 1 cvičení), v ScheduleBox se u tohoto předmětu zobrazí pouze tyto zapsané akce, s možností jejich odzápisu.
    - Pokud podmínky zápisu pro předmět splněny nejsou (např. chybí jedna přednáška), ScheduleBox zobrazí pro tento předmět (zejména pro chybějící typ akce) všechny dostupné rozvrhové akce daného typu, aby si uživatel mohl vybrat a doplnit chybějící zápis.
- **Detail akce:** Kliknutím na komponentu rozvrhové akce (ScheduleEvent.jsx) v rozvrhu (ScheduleBox.jsx) se zobrazí ScheduleEventDropdown s dalšími informacemi.

5.4.7. Export a Tisk Rozvrhu

- Možnost uložit vygenerovaný rozvrh jako rastrový obrázek (např. .png) pomocí WorkspaceService.saveScheduleImage().
- Rozvrh bude možné vytisknout.

5.4.8. Kontrola vstupů

Aplikace by měla provádět kontrolu vstupů, aby do formulářových polí nešly zadat nesmyslné hodnoty a informovat uživatele o problémech. Toto je relevantní např. při vytváření preferencí nebo manuálním zadávání předmětů/akcí.

5.4.9. Podpora Vícejazyčnosti

Celá aplikace bude podporovat český a anglický jazyk. To zahrnuje texty v uživatelském rozhraní, popisky, chybové hlášky a další textový obsah.

6. Technologie

- **Framework/Knihovna:** React + Vite.
- **UI Komponenty:** MUI (Material-UI) a MUI-X-Treeview pro stromové zobrazení předmětů. Použití komponent třetích stran je v souladu s požadavky. SwipeableDrawer a Tab komponenty z MUI budou použity pro mobilní responzivní design.
- **Stylování:** Vzhled aplikace může být definován v odděleném CSS souboru. Aplikace bude mít soubor theme.jsx pro definici a customizaci MUI tématu.
- **Barevná schémata:** Aplikace bude podporovat světlý (white mode) a tmavý (dark mode) režim zobrazení, spravovaný přes theme.jsx.
- **Správa stavu:** Využití principů podobných Observable-Observer (React hooks) a data bindingu.

Tato dokumentace by měla sloužit jako solidní základ pro vývoj aplikace a splňuje požadavek na detailní popis pro LLM.