Podmienky pre odovzdanie semestrálnej práce z predmetu VAII 2025/26

Pre absolvovanie predmetu *vývoj aplikácií pre internet a intranet* (VAII) je potrebné vypracovať semestrálnu prácu. Požiadavky, ktoré práca musí spĺňať, sú uvedené v nasledujúcom texte.

Kontrolné termíny a semestrálna práca

Počas semestra bude stav semestrálnej práce kontrolovaný v dvoch termínoch a to v **7.** a **12.** týždni. Na kontrolnom termíne a obhajobe semestrálnej práce sa bude hodnotiť splnenie týchto požiadaviek:

- **Termín 1 (max. 10 bodov)**
  Vytvorenie dokumentu, ktorý bude popisovať Vašu semestrálnu prácu podľa priloženej šablóny.
  - Text pokrýva všetky požadované časti (úvod, prehľad podobných aplikácií, analýza, návrh).
  - Každá kapitola je primerane rozpracovaná, nechýbajú dôležité informácie.
  - Informácie sú vecné, aktuálne a logicky podané.
  - Text je napísaný spisovne, bez slangu, pravopisných a štylistických chýb.
  - Používa sa odborný jazyk primeraný vysokoškolskému prostrediu.
  - Je dodržaná štruktúra dokumentu uvedená v šablóne.

  Pozor! Navrhované riešenie **musíte aj realizovať**. Nie je možné vypracovať semestrálnu prácu na inú tému, ako ste uviedli v dokumente. Takisto rozsah aplikácie a funkcie, ktoré navrhnete, by ste mali skutočne v práci aj implementovať.

- **Termín 2 (max. 10 bodov)**
  - Uloženie projektu v GIT
  - Aspoň 10 vlastných CSS pravidiel (externe pripojené)
  - Implementované základne rozloženie webu (layout), responzívny dizajn
  - Implementované všetky CRUD operácie na strane servera, včítane grafického rozhrania k nim
  - Kontrola formulárových prvkov na strane klienta
  - Kontrola formulárových prvkov na strane servera
  - Netriviálny JavaScript

  Pozor! Odovzdané riešenie nesmie byť kópiou, alebo len upravenou verziou príkladov z cvičení. Pre povolenie odovzdania semestrálnej práce je nutné, aby študent získal **z každého** z kontrolných termínov **min**. **6 bodov**.

- **Semestrálna práca (max. 58 bodov)**
  - Správne použitie git
  - Kvalita DB návrhu
  - Kontrola všetkých používateľských vstupov na strane servera
  - Zabezpečenie aplikácie
  - 2x zmysluplné použite AJAX volania
  - Aplikácia používa viacero rolí + autorizácia
  - Práca so súbormi (*upload*, manažment súborov)
  - Kvalita návrhu architektúry aplikácie a programovací štýl
  - Zložitosť aplikácie, množstvo funkcií, prepracovanosť
  - Výsledný dojem z aplikácie

- **Body za nadštandardnú semestrálnu prácu (max. 12 bodov)**
  - Použitie nejakého frameworku vo vhodnom rozsahu (okrem FW Vaííčko)
  - Prepracovaný dizajn aplikácie
  - Použitie nejakého JS frameworku
  - Vlastné riešenie responzívneho dizajnu
  - Použitie *less*, *sass*, pokročilých vlastností CSS
  - Použitie docker (migrácia a vloženie dát)
  - Možnosť jednoduchého nasadenia aplikácie (migrácie, *seeding* dát)
  - Použitie externých API
  - Implementácia niektorého z HTML5 API
  - Vlastná pokročilá architektúra aplikácie => vlastný framework, MVC, ...
  - Pokročilá kontrola formulárov na strane klienta

Termíny obhajoby semestrálnej práce

1. Riadny termín odovzdania práce bude vopred oznámený a na ňom je možné získať **70 bodov**. Termín býva zvyčajne v 1. polovici januára, ale prácu je možné odovzdať aj v priebehu cvičení v 12. týždni.
2. Náhradný termín bude cez skúškové obdobie (zvyčajne 2. polovica januára), na tomto termíne je možné za semestrálnu prácu získať max. **48 bodov**. Na tomto termíne sa už **neudeľujú** body z časti „nadštandardná semestrálna práca".

Povinné náležitosti semestrálnej práce

Semestrálna práca musí spĺňať nasledovné **povinné požiadavky**. Ak niektoré z nich nie sú splnené, prácu nie je možné obhajovať.

Ak máte špecifickú prácu, ktorá nespĺňa tieto požiadavky, ale je dostatočne zložitá v inom smere (napr. netriviálna hra v JS a pod.), je nutné sa **vopred** s vyučujúcimi dohodnúť, či je práca vhodná ako semestrálna.

Povinné požiadavky na semestrálnu prácu:

- aplikačná logika oddelená od prezentačnej vrstvy
- min. 5 dynamických stránok
- min. 50 vlastných riadkov kódu v Javascripte (Typescript, Dart, ...)
- min. 20 vlastných CSS pravidiel (mimo Bootstrap a pod.)
- min. 3 zmysluplné DB entity, všetky použité v aplikácii (tabuľka používateľov, napr. ***entita users*** **sa** do týchto entít **nezapočítava**)
- aspoň jeden 1:N alebo M:N vzťah medzi DB entitami
- implementované **všetky CRUD** (čítanie, vytváranie, úprava a mazanie dát) operácie nad dvomi entitami
- aplikácia musí obsahovať časť, do ktorej je nutné sa prihlásiť
- k aplikácii je nutné priložiť **návod** na inštaláciu (napr. súbor README.md)

Do počtu sa nerátajú prevzaté (napr. z cvičení alebo zbierky úloh z VAII) a mierne upravené, frameworkom/knižnicou vygenerované DB entity, stránky, CSS pravidlá, kód, atď.

Semestrálna práca

1. Semestrálna práca musí byť **webová aplikácia**, ktorá využíva nejaké dynamické technológie na strane servera.
2. Každý študent vypracováva prácu **samostatne** a **osobne** ju aj obhajuje, pričom ju nie je možné odovzdať prostredníctvom iného študenta. Pri vypracovaní práce môžete v neobmedzenej podobe používať nástroje generatívnej AI (ChatGPT, GitHub copilot, ...). **Odovzdávanej práci musíte dokonale rozumieť a byť pripravený odpovedať na doplňujúce otázky. V prípade, že práci nerozumiete a neviete odpovedať na doplňujúce otázky, je to dôvod na ukončenie obhajoby a udelenie známky Fx.**
3. Všetky časti semestrálnej práce, ktoré ste vygenerovali s AI, je nutné mať riadne označené.
4. Téma práce musí byť určená vopred v dokumente odovzdávanom pri kontrolnom termíne. Zmena témy je možná vo výnimočných prípadoch len po dohode s vyučujúcim.
5. Jednu semestrálnu prácu môžu vypracovať aj **viacerí študenti** spoločne. Zložitosť takejto práce musí byť adekvátna počtu študentov, pričom každý študent pracuje na svojej časti, avšak musí spĺňať všetky podmienky na celú semestrálnu prácu.

Rozdelenie práce medzi študentov musí byť jednoznačne stanovené. Pred odovzdaním takejto práce je potrebné prebrať tému práce a jej rozdelenie s vyučujúcim.

6. Prácu je možné prezentovať na vlastnom počítači alebo notebooku, prípadne umiestniť na verejne dostupný webhosting. Zdrojové kódy práce musia byť pre hodnotiaceho **dostupné cez GIT repozitár**. Práca musí byť vypracovaná postupne a v repozitári musí byť vidieť postupný progres od začiatku semestra.
7. Práca sa obhajuje **priamo** pri odovzdávaní, kde študent obháji svoje riešenie semestrálnej práce. Študent musí **mať prístup** ku zdrojovým kódom aplikácie počas obhajoby práce, aby bolo možné overiť samostatnosť vypracovania.
8. Ak práca používa kód, ktorého študent nie je autorom, je potrebné k tomuto kódu pripísať do komentára jeho zdroj.

Aplikácia

Výsledná internetová aplikácia je rozdelená na **klientsku a serverovú** časť, ktoré musia spĺňať nasledovné požiadavky:

Všeobecné požiadavky

1. Zdrojový kód musí byť správne štruktúrovaný, členený a prehľadne formátovaný. Dôvodom na zrážku bodov je neprehľadný, zle členený alebo často sa opakujúci kód.
2. Pri vytváraní by ste mali dodržiavať postupy objektovo orientovaného programovania (*OOP*) a používať *MVC (prípadne iné architektúry ako MVVM, MVP, ...).*
3. Hodnotí sa kód, ktorý je vytvorený študentom. Pokiaľ je v aplikácii použitý framework, CMS alebo iné hotové riešenie, bude hodnotená miera modifikácie alebo implementácie, ktorú študent sám vytvoril. V odovzdávanej aplikácií však musí byť v dostatočnej miere obsiahnutá aj **vlastná tvorivá činnosť** študenta. Ak chcete odovzdať semestrálnu prácu tohto typu, je vhodné svoj zámer vopred prekonzultovať s vyučujúcim.
4. Aplikácia musí dodržať tieto zásady bezpečnosti:
   a. Kontrola vstupov, kontrolujúca **nielen dátový typ**, ale aj **platnú hodnotu** na strane servera aj klienta.
   b. Ošetrenie dopytov na DB voči útokom *SQLInjection*.
   c. Aplikácia musí obsahovať časť prístupnú až **po prihlásení používateľa** (t.j. musí obsahovať prihlasovací formulár).
   d. **Heslá** užívateľov nesmú byť uložené ako obyčajný text.
   e. Ak je aplikácia typu SPA, koncové body (*endpoints*) musia byť zabezpečené tak, aby nebolo možné na ne pristúpiť neautorizovaným spôsobom.
5. Klientska strana musí so stranou servera komunikovať **asynchrónne** (**AJAX**), min. dvoma spôsobmi z nasledujúceho zoznamu:
   a. odosielanie formulárov
   b. editovanie záznamu priamo v tabuľke (*in-place editing*)
   c. prihlasovanie do aplikácie
   d. filtrovanie záznamov tabuľke
   e. načítavanie obsahu tabuľky
   f. stránkovanie obsahu

g. iné netriviálne volania, podľa vlastného výberu

Server – databáza

1. Musí obsahovať **min. 3 entity** (databázové tabuľky). Do počtu sa nezarátava asociačná tabuľka (dekompozícia vzťahu M:N) a frameworkom / knižnicou vygenerované tabuľky a ani tabuľka používateľov.
2. Databáza by mala spĺňať **normalizáciu**; t.j. žiadne duplicity dát a nezmyselné tabuľky, ktoré sú vytvorené len do počtu. Príklady nevhodného použitia napr.:
    a. *Pre každého užívateľa sa generuje vlastná tabuľka – stačí jedna tabuľka, používateľov rozdeľuje ich ID atribút*
    b. *Každá rola užívateľa má vlastnú tabuľku – do tabuľky používateľov stačí pridať stĺpec, ktorý rolu určuje*

Server – aplikačná logika

1. Aplikácia vykonáva nad DB všetky operácie – **vkladanie, čítanie, mazanie a upravovanie dát.**
2. Aplikácia využíva všetky entity v DB. Nepoužívaná entita sa nezarátava do počtu povinných entít.

Klient – aplikačná logika + GUI

1. Aplikácia musí obsahovať **min. 5 rôznych podstránok** s dynamickým obsahom.
2. Aplikácia musí mať **responzívny dizajn**, t.j. jej zobrazenie sa musí vhodne prispôsobiť rozlíšeniu displeja, na ktorom bude zobrazovaná.
3. Vzhľad a formát aplikácie musí definovať minimálne **20 CSS pravidiel.**
4. Je potrebné vypracovať vlastný klientsky skript o veľkosti **minimálne 50 riadkov**. Skript musí byť zmysluplný a používaný v aplikácii.
5. Používateľské rozhranie musí spĺňať pokročilé funkcie pre správu dát v aplikácii. Nesprávne riešenie je napr. mazanie záznamu pomocou manuálnom zadaní ID záznamu a pod.