**ВЕБ ПРОГРАМИРАЊЕ**

**Циљ** учења предмета Веб програмирање је да ученици, кроз упознавање са савременим веб- технологијама и њихово коришћење у циљу креирањa веб-садржаја који одговарају савременим пословним и личним потребама корисника, развију компетенције за рад са подацима и одговорно коришћење информационо-комуникационих технологија, као и да ученике оспособи за примену усвојених знања из области рачунарства и информатике, решавање разноврсних проблема из животне праксе, да их припреми за наставак образовања, да допринесе развијању менталних способности, формирању научног погледа на свет и свестраном развитку личности ученика.

ОПШТА ПРЕДМЕТНА КОМПЕТЕНЦИЈА

Учењем наставног предмета Веб програмирање ученици су оспособљен да примене стечена знања и вештине из области информационо-комуникационих технологија ради испуњавања постављених циљева и задатака у свакодневном животу, даљем школовању и будућем раду. Ученици развијају способност апстрактног и критичног мишљења уз помоћ информационо-комуникационих технологија. Развијају дигиталну писменост и позитивне ставове према рачунарским наукама.

СПЕЦИФИЧНЕ ПРЕДМЕТНЕ КОМПЕТЕНЦИЈЕ

Специфичне предметне компетенције представљају опис специфичних способности ученика које му омогућавају да развије општу предметну компетенцију. Ученици одговорно користе информационо-комуникационе технологије уз препознавање потенцијалних ризика и опасности. Брзо, ефикасно и рационално проналазе информације коришћењем рачунара, критички их анализирају и представљају у базама података. Познају концептe веб програмирања и језике за опис садржаја, изгледа и понашања веб страна. На основу познавања језика HTML и CSS, тумаче елементе веб-странице, прилагођавају их и креирају визуелно допадљиве странице које садрже линкове, слике, листе, табеле и сличне елементе. Овладали су вештинама и техникама неопходним за креирање сложенијих вишеслојних веб–апликација које омогућавају кориснику да кроз формулар уноси одређене податке, добије жељене податке, претражује, ажурира или брише податке из базе података.

|  |  |
| --- | --- |
| Разред | **Четврти** |
| Недељни фонд часова | **2 часа вежби** |
| Годишњи фонд часова | **66 часова вежби** |

|  |  |
| --- | --- |
| **ИСХОДИ**  По завршетку разреда ученик ће бити у стању да: | **ТЕМА**  и кључни појмови садржаја програма |
| * објасни намену и структуру савремених рачунарских мрежа; * објасни појмове слојева и протокола; * разликује улоге појединих слојева; * разликује начине адресирања на различитим мрежним слојевима; * разликује улоге појединих хардверских уређаја и комуникационе опреме у мрежној комуникацији; * разликује кључне моменте у развоју интернета; * идентификује функције интернета; * објасни намену различитих интернет - сервиса; * разликује начине адресирања на интернету. * форматира текст у HTML документу; * креира листе и табеле у HTML документу; * повезује више HTML страница и мултимедијалних садржаја; * креира статичку веб-страну коришћењем описног језика HTML; * креира веб-страницу која садржи формуларе. * објасни везу између језика HTML и језика стилских листова CSS; * примењује својства и селекторе у CSS документу: * распоређује елементе веб-странице применом језика CSS; * стилизује веб-страницу написану језиком HTML; * објасни основну намену језика JavaScript; * угради скрипт у веб-страницу; * користи основне елементе језика JavaScript; * користи JavaScript објекте; * примењује функције у JavaScript коду; * обради податке унете преко формулара на веб-страници коришћењем језика Java Script; * креира веб-страницу са интерактивним елементима; * разликује принципе израде статичких, активних и динамичких веб-страна; * објасни улоге веб-клијената и веб-сервера; * користи основне елементе језика и програмског интерфејса за креирање серверских скрипти; * развије једноставну веб-апликацију; * креира динамички веб-сајт; * креира серверски скрипт који прихвата и обрађује податке унете у формулар на веб-страници; * формира садржај веб-стране на основу података; * изврши упит из скрипта према бази података; * тимски и уз помоћ наставника дефинише проблем за чије решење се користи динамички веб-сајт; * тимски развије и прикаже идејно решење проблема; * тимски развије план рада и начин праћења успешности реализације плана; * креира динамички веб-сајт за дефинисан проблем; * креира, уређује и структурира садржаје током рада на пројекту; * пише документацију; * креира презентацију и презентује решење пројектног рада; * вреднује своју улогу при изради пројектног задатка и активности за које је био задужен. | **РАЧУНАРСКЕ МРЕЖЕ И ИНТЕРНЕТ**  Обнављање и систематизација знања о рачунарским мрежама и интернету стечених у ранијем разредима из предмета Примена рачунара и Оперативни системи и рачунарске мреже:   * Појам и сврха рачунарске мреже. * Компоненте рачунарских мрежа. * Улога уређаја у мрежној комуникацији. * Мрежни хардвер и софтвер. * Интернет сервиси. * Интернет протоколи. |
| **ОПИСНИ ЈЕЗИК HTML**   * Универзални принципи веб дизајна и готова веб дизајн решења (WordPress, Weebly, Wix…). * Основе језика HTML, основни елементи и атрибути. * Структура HTML стране. * Заглавље веб-стране и метаподаци. * Форматирање текста. * Листе и табеле. * Везе и адресе. * Уметнути мултимедијални садржај. * Формулар. |
| **СТИЛСКИ ЛИСТОВИ – ЈЕЗИК CSS**   * Увод у језик стилских листова CSS. * Укључивање стилских листова у HTML документе. * Најчешће коришћена својства и њихове вредности. * Селектори. * Распоред елемената. * Пример стилизовања веб-стране. |
| **СКРИПТ ЈЕЗИК JAVASCRIPT ЗА КЛИЈЕНТСКО ПРОГРАМИРАЊЕ**   * Уметање скрипта у веб-странице. * Извршавање скрипта. * Основе језика JavaScript: вредности, променљиве, изрази, објекти и низови, елементарни типови и оператори. * Функције, методе, догађаји. * Контрола тока програма. * Израда формулара у веб страници * Убацивање интерактивних елемената у веб страницу коришћењем језика Java Script |
| **СЕРВЕРСКО ПРОГРАМИРАЊЕ**   * Улога клијената и сервера у веб-апликацијама. * Основни елементи одабраног језика за креирање серверских скрипти (променљиве, изрази, наредбе, структуре података). * Пренос података између клијената и сервера. * Обрада података коришћењем серверских скриптова. * Динамичко генерисање веб-странице коришћењем скриптова. * Програмски интерфејс према базама података и употреба база података у веб-апликацијама. |
| **ПРОЈЕКТНИ ЗАДАТАК**   * Фазе пројектног задатка од израде плана до представљања решења. * Израда пројектног задатка. * Вредновање резултата пројектног задатка. |

**УПУТСТВО ЗА ДИДАКТИЧКО-МЕТОДИЧКО ОСТВАРИВАЊЕ ПРОГРАМА**

Настава вежби се изводи са половином одељења у рачунарском кабинету, у групама не већим од 12 ученика.

I ПЛАНИРАЊЕ НАСТАВЕ И УЧЕЊА

Приликом планирања часа, исходе предвиђене програмом треба разложити на мање и на основу њих планирати активности за конкретан час. Треба имати у виду да се исходи у програму разликују, да се неки могу лакше и брже остварити, док је за одређене исходе потребно више времена, активности и рада на различитим садржајима. Исходе треба посматрати као циљеве којима се тежи током једне школске године.

При обради нових садржаја треба се ослањати на постојеће искуство и знање ученика, и настојати, где год је то могуће, да ученици самостално откривају математичке правилности и изводе закључке. Ученике треба упућивати да користе уџбеник и друге изворе знања, како би усвојена знања била трајнија и шира, а ученици оспособљени за примену у решавању разноврсних задатака.

На часовима треба комбиновати различите методе и облике рада, што доприноси већој рационализацији наставног процеса, подстиче интелектуалну активност ученика и наставу чини интересантнијом и ефикаснијом. Препоручује се коришћење интерактивних метода, пројектне, проблемске и истраживачке методе, дискусије, дебате и др., како би ученици били што више ангажованом током наставе. Комбиновати на часовима различите облике рада као што су самостални рад ученика (по принципу један ученик – један рачунар), рад у паровима (два ученика истовремено и заједно решавају конкретне задатке), рад у мањим групама (почетна анализа и идеје за методе решавања), као и рад са целом групом када наставник објашњава, приказује, демонстрира и кроз дискусију уводи ученике у нове области. Избор метода и облика рада, као и планирање активности ученика ускладити са наставним садржајем који треба реализовати на часу и предвиђеним исходима, али и са специфичностима одељења и индивидуалним карактеристикама ученика. Предложени број часова по темама је оквирни, на наставнику је да процени потребан и довољан број часова по темама узимајући у обзир знања и вештине који ученици имају из претходног школовања и животног искуства. Предложени редослед тема није обавезујући за наставнике, већ само представља један од могућих модела, који наставник може прилагодити у складу са изабраним програмским језиком и методолошким опредељењем.

Ради лакшег планирања наставе даје се оријентациони предлог броја часова по темама.

* Рачунарске мреже и интернет (6 часова)
* Описни језик HTML (10 часова)
* Стилски листови – језик CSS (10 часова)
* Скрипт језик JavaScript за клијентско програмирање ( 15 часова)
* Серверско програмирање ( 15 часова)
* Пројектни задатак ( 10 часова)

НАПОМЕНА: Предвиђена су два двочасовна писмена задатка са исправком (6 часова). Наставник бира како се ради писмени задатак, писано на папиру или на рачунару. У случају да се ради на рачунару, радови ученика могу да се чувају у електронском облику.

### II. ОСТВАРИВАЊЕ НАСТАВЕ И УЧЕЊА

У оквиру тема **Рачунарске мреже и интернет** потребно је:

Са ученицима обновити и систематизовати знања о рачунарским мрежама и интернету стечених у ранијем разредима из предмета Примена рачунара и Оперативни системи и рачунарске мреже.

Дефинисати појам рачунарске мреже и приказати основне класификације мрежа. Описати основне мрежне уређаје и комуникационе канале (жичане и бежичне) у савременим рачунарским мрежама. Објаснити појам архитектуре мреже, објаснити појам слојевитости мреже и мрежних протокола и навести најчешће коришћене протоколе и технологије преноса података. Посебно објаснити начине адресирања и рутирања (нарочито појмове *IP* адресе, *URL*, *DNS* и друге битне за разумевање области Серверско програмирање). Укратко упознати ученике са протоколом TCP (описати предности комуникације поделом порука на пакете и механизам потврде пријема којим се обезбеђује поуздана комуникација). Описати употребу и функционисање апликативних протокола за веб (*HTTP*, *HTTPS*) и за електронску пошту (POP3, IMAP, SMTP). Фокусирати се на практичне аспекте поменутих протокола (подешавање клијента електронска поште). Обновити појам интернет сервиса.

У оквиру теме **Описни језик HTML** потребно је да:

Oбјаснити ученицима појам веба (енгл. world wide web, WWW), поделу на статички и динамички веб, поделу на клијентске и серверске технологије и теорију веб дизајна (шта је веб дизајн, аспекти веб дизајна, технологије веб дизајна, развој веб сајта). Објаснити разлику између статичке веб стране, динамичке веб стране која може да садржи формуларе за спрегу са неком базом података и веб портала (дати примере конкретних портала са којима се ученици срећу, попут школског електронског дневника, портала за електронско пословање или портала који се користи у забавне сврхе). При налажењу примера на интернету пожељно је поделити ученике у тимове ради лакшег и бржег проналажења ових примера у складу са интересовањима ученика. Код поделе на статички и динамички веб подстаћи ученике да сами пронађу примере претраживањем на интернету. Продискутовати евентуалне предности и мане оба приступа.

Потребно је поменути, без улажења у детаљнији опис програмских језика, најчешће клијентске технологије (HTML, CSS, Java Script) и најчешће серверске технологије (PHP, Python, Ruby, ASP.Net/C#, NodeJS...).

Представити језик HTML (Hypertext Markup Language) који служи за означавање хиптерекста и хипермедија (текста, слике, звука, видеа…), међусобно повезаних објеката помоћу линкова. Истаћи постојање различитих верзија стандарда језика и приликом излагања се држати искључиво најновијег стандарда.

На примеру готове стране приказати могућност приказа HTML кода унутар прегледача веба. Приказати могућности које савремени прегледачи веба нуде креаторима веб-страница (обично је то опција F12, web developer tools) и приказати како те алатке помажу да се идентификују појединачни елементи веб-страница и њихов опис у језику HTML.

Кроз примере увести општу синтаксу језика HTML. Објаснити појам елемента у језику HTML, тагове (етикете), и њихову особину да могу бити „упарени” или „неупарени” тј. „затварајући” и „самозатварајући”. Објаснити да се елемент у језику HTML састоји од отварајућег тага, садржаја и затварајућег тага. Објаснити појам атрибута, запис атрибута у оквиру тага као и то да атрибути увек описују неки елемент и не могу се писати самостално. Нагласити да атрибути најчешће прецизније дефинишу начин на који се елементи приказују у прегледачу. Објаснити да неки атрибути нису обавезни, али су некада врло битни и пожељни. Нагласити да да су имена атрибута, њихове вредности, тип и интервали унапред дефинисани језиком и да различити елементи могу бити описани различитим атрибутима.

Представити елементе који описују основну структуру HTML документа (<html>, <head>, <title>, <body>). Објаснити елемент мета којим се задају основне мета-информације о документу. Истаћи употребу атрибута charset и његову везу са кодирањем карактера приликом чувања документа. Приказати примере HTML докумената чији је садржај записан на ћирилици и на латиници и показати како се обезбеђује њихов исправан приказ.

Представити основне елементе за организацију текста унутар веб-странице: елементе за обележавање наслова (<h1>, <h2>, ...), елементе за обележавање пасуса (<p>) и елементе за обележавање листа са нумерацијом и без нумерације (<ul>, <ol>, <li>) и инсистирати на томе да ученици добро овладају коришћењем ових основних елемената. Поменути и семантичке елементе за организацију садржаја странице (<header>, <footer>, <nav>, <main>, <section>, <article>, ...), поменути и друге, ређе коришћене елементе (на пример, <address>, <blockquote>, <code>, ...), али не инсистирати да их ученици напамет уче. Скренути пажњу ученицима на могућност коришћења документације и референтних прегледа и приручника.

Описати елементе за дефинисање табела (<table>, <tr>, <th>, <td>). Навести најзначајније атрибуте којима се ови елементи описују (width, height, border, rowspan и colspan за спајање ћелија итд.).

Описати елемент <a> и атрибут href за креирање хиперлинкова. Подсетити ученике на појам URL и описати референцирање објеката путем релативне и путем апсолутне адресе. Описати креирање линкова ка деловима унутар веб-стране.

Описати основне елементе за промену физичког и логичког стила карактера (<b>, <i>, <u>, <emph>, <strong>, ...)

Описати елементе за уметање мултимедијалног садржаја у HTML странице: <img> за уметање слика, <video> за уметање видео-записа и <audio> за уметање аудио-записа. Навести значај атрибута alt, значај компресије и прилагођавање формата мултимедијалног садржаја за коришћење на вебу, атрибуте width и height и слично.

Описати основне елементе за опис формулара у склопу веб-страница и нагластити да ће се детаљно радити обрада формулара у склопу теме писања клијентских и серверских веб-скриптова.

У оквиру теме **Стилски листови - језик CSS** потребно је да:

Објаснити улогу CSS (Cascading Style Sheets) стилова код визуелног стилизовања HTML страна. Истаћи значај јасног разликовања логичког описа садржаја стране помоћу језика HTML и описа њене визуелне презентације помоћу језика CSS.

Описати начине да се елементу измени стил: коришћење атрибута style, коришћење елемента <style> у заглављу веб-странице и коришћење екстерног стилског описа увезеног у веб-страницу. Описати када је пожељно користити од ове начине стилизовања (стилизовање на нивоу веб-сајта, стилизовање на нивоу веб-странице), као и приоритет различитих стилских описа.

Описати основне селекторе у језику CSS: селекцију на основу назива елемента, селекцију на основу идентификатора елемента (оператор #), селекцију на основу класе елемента (оператор .), селекцију угнежђених елемената (нпр. p img селектује све слике које се налазе унутар пасуса), комбиновање селектора (нпр. h1, h2 селектује све наслове првог и све наслове другог нивоа).

Описати основна својства елемената и њихове вредности: својства која се односе на фонт и атрибуте текста, својства која се односе на боју текста и боју позадине, својства која се односе на ширину и висину елемената, својства која се односе на оквире елемената и својства која се односе на унутрашње и спољашње маргине.

Поменути употребу језика CSS за распоређивање садржаја на страници (својства која се односе на позиционирање елемената и на плутајуће елементе тј. својство float).

При реализацији ове тематске целине потребно је све странице креирати у текстуалном едитору, при чему то може бити едитор прилагођен креирању веб-страница, који нуди бојење кода на основу синтаксе, аутоматско допуњавање започетог кода, падајуће листе за избор HTML елемената и атрибута и слично. Подстаћи ученике да примене стечена знања из рада са програмима за обраду текста и програмима за обраду слика и тако припреме садржај за креирање сопствене веб-странице. При бирању садржаја инсистирати на критичком приступу информацијама и и неговању естетике.

У оквиру теме **Скрипт језик JavaScript за клијентско програмирање** потребно је:

Језик JavaScript омогућава да се веб-страницама дода интерактивност и представља де факто стандард у писању скриптова на страни веб-клијента. У склопу наставе програмирања нема простора да се изуче све функционалности овог језика, али и елементарно упознавање ученика са функционалношћу коју клијентски скриптови омогућавају може учинити наставу веб-програмирања интересантнијом и кориснијом за будуће потребе ученика.

Често веб-апликације имају кориснички интерфејс креиран коришћењем формулара и контрола (поља за унос текста, натписа, дугмића, листа и слично) описаних у језику HTML. Пре описивања језика JavaScript (или паралално са његовим увођењем) ученицима описати HTML елементе намењене креирању формулара (form, input, label, textarea, ...). Приказати већ готову форму за логовање на некој HTML страни и на њој објаснити <input /> таг и најчешће коришћене елементе форме: текстуално поље за унос, check поља и тастере. Објаснити и друге елементе који се користе у формама: radio тастери и dropdown листе.

Током изучавања елементарних појмова језика, са ученицима је могуће радити задатке специфичне за уводну наставу програмирања (који нису тесно везани за веб-програмирање), који покривају изразе, гранање, петље, рад са колекцијама података (пре свега низовима) и писање функција (уз евентуално илустровање концепта анонимних функција). Ако током претходних година ученици нису изучавали неки језик са синтаксом преузетом од програмског језика C, током наставе је пожељно детаљније обрадити синтаксичке аспекте језика (на пример, петљу for са иницијализацијом, условом и кораком, употребу витичастих заграда за означавање блокова и слично). Ако су током претходних година ученици изучавали неки статички-типизиран језик, потребно је им је објаснити концепт динамичке типизираности (која се користи у језику JavaScript).

Основна употреба језика JavaScript долази у облику приступа елементима веб-страница кроз објектни модел документа (енгл. Document Object Model, DOM), који омогућава да се елементима веб-странице приступа путем објеката у програму. Најједноставнији механизам је приступ елементу на основу идентификатора (функцијом document.getElementById) и велики број примера је могуће засновати на њему. Након што се добије објекат којим је елемент представљен најинтересантније је прочитати или променити његов садржај (на пример, коришћењем поља innerHTML или value), променити му атрибуте (нпр. атрибут src елемента img, чиме се мења слика) или стил (коришћењем поља style или class), сакрити или приказати неки елемент. DOM нуди заиста богат програмски интерфејс (на пример, могуће је приступати елементима прозора, веб-прегледача и слично), међутим, нема ни простора, а ни потребе да се он у потпуности илуструје. Ученицима скренути пажњу на могућност коришћења документације и подстицати их да уместо да сву функционалност уче напамет, током програмирања користе документацију.

Још један важан аспект програмирања апликација са графичким корисничким интерфејсом, па и веб-апликација чини концепт догађаја и обрада догађаја. Најједноставнија могућност реаговања на догађаје је навођење програмског кода као вредности специјалних атрибута догађаја (на пример, onclick), но он се данас не сматра пожељним (мада се због једноставности може користити у почетним примерима). Бољи начин је регистровање функција за обраду догађаја (регистрација се врши позивом document.addEventListener). Скренути пажњу ученицима на асинхрону природу овог механизма (иако се регистрација врши одмах, функција за обраду догађаја се позива асинхроно, у будућности, када догађај наступи).

Једна група примера треба да илуструје опште механизме програмирања графичког корисничког интерфејса, са којима се ученици још нису срели (на пример, програм који сабира бројеве унете у поља за унос текста након што се кликне на дугме или одмах након што се измени садржај неког текст поља или програм који на основу података уписаних у мало богатији формулар за наручивање неког производа формира текст наруџбенице). Могући интересантни примери су програмирање једноставне галерије слика, програмирање сакривања и приказивања делова веб странице у циљу њеног лакшег прегледања, увећавање величине слова у неком пасусу дуплим кликом на њега, како би се текст могао лакше прочитати, приказ дигиталног сата у неком елементу, промена слике другом сликом једним кликом на дугме (симулација паљења сијалица) и слично.

Ако наставник то сматра корисним, било у склопу редовне, било у склопу додатне наставе у програмима је могуће користити и неке специјализоване JavaScript/CSS библиотеке (на пример, JQuery, Bootstrap, ...), као и рад са дводимензионалном графиком (Canvas API) и слично.

У оквиру теме **Серверско програмирање** потребно је:

За разлику од клијентског веб-програмирања где се језик JavaScript усталио као де факто стандард, серверско веб-програмирање се може илустровати коришћењем различитих програмских језика. Једна од могућности је да се употреби језик PHP који је специјализовани језик за серверско веб-програмирање и даље је заступљен на највећем броју веб-сајтова (пре свега захваљујући употреби унутар система WordPress). Ако су ученици током школовања користили програмски језик Python, могуће је користити и његове специјализоване радне оквире за креирање веб-апликација (такви су, на пример, Flask или Django). С обзиром на то да су ученици већ упознати са програмирањем клијентских скриптова коришћењем језика JavaScript, још једна могућност је да се серверски скриптови програмирају коришћењем платформе node.js. Ако су ученици изучавали програмски језик C# или Java, могуће је користити њихове специјализоване радне оквире за програмирање веб-апликација (ASP.net или JSP). Овај списак није исцрпан и наставник има слободу да одабере било који програмски језик тј. радни оквир који омогућава достизање прописаних исхода.

Ако је одабран програмски језик који ученици нису раније користили, описати и кроз низ веома једноставних примера илустровати релевантне аспекте његове синтаксе (запис израза, наредбе, колекције података, дефинисање функција и слично).

Серверски скриптови често примају податке који се уносе у веб-формуларе и који се скриптовима преносе преко HTTP протокола (најчешће методом GET или POST). Кроз примере формулара описати релевантне аспекте тог протокола и приказати како се из скрипта приступа пренетим подацима (за праћење HTTP захтева и одговора могуће је користити функционалности савремених прегледача веба намењених развоју софтвера). Описати разлику између метода GET и POST и описати случајеве у којима је адекватно користити један и случајеве у којима је адекватно користити други метод.

Типичан резултат серверских скриптова је генерисана веб-страница, описана у језику HTML. Илустровати овај концепт кроз примере (на пример, написати серверски скрипт који генерише HTML страницу која садржи таблицу множења или таблицу вредности неке функције). Ако одабрани програмски језик и радни оквир који се користи омогућавају креирање HTML страница коришћењем шаблона, илустровати тај концепт кроз једноставне примере.

Вероватно централна функционалност серверских скриптова је њихова могућност повезивања са базом података и могућност читања, уписа, измене и брисања података у базама. Кроз низ адекватно одабраних примера приказати ученицима ове могућности. Приказати поступак повезивања скрипта са базом података, извршавања упита (са и без параметара) и обраде резултата упита. На пример, ако база садржи податке о ученицима и њиховим оценама, скриптови могу да имплементирају основне функционалности електронског дневника (на пример, приказ оцена свих ученика из једног предмета, упис нове оцене на контролном задатку, измена података о појединачном ученику, пребацивање свих ученика у наредни разред, брисање неке оцене и слично). Овој теми посветити значајан број часова и током обраде ове теме инсистирати на детаљном увежбавању одабраног подскупа упитног језика SQL. Скренути пажњу ученицима на опасности од извршавања упита сачињених од вредности које корисник уноси уместо да се користи параметризован упит (напад SQL уметања, тј. SQL injection) и неке савете да се ти проблеми превазиђу.

Осим табеларног приказа резултата, скриптови се често користе и да генеришу формуларе унутар веб-страница. На пример, приликом уписа оцена у електронски дневник, приказује се падајући мени за избор ученика или предмета чији је садржај аутоматски генерисан скриптом, на основу података из базе. Илустровати и користити и ову могућност.

Протокол HTTP не чува стање и апликације које чувају стање морају користити сесије или колачиће. Иако ови концепти превазилазе основни ниво серверског веб-програмирања, ако одабрани програмски језик нуди њихово једноставно коришћење, наставник може одабрати да ученицима кроз неколико примера прикаже њихову употребу. Могуће је, на пример, реализовати механизам логовања на сајт.

Осим враћања форматираних страница у језику HTML, серверски скриптови данас често враћају чисте податке (често у формату JSON или XML), који се онда обрађују, форматирају и приказују на страни клијента (клијентским JavaScript скриптовима). Ово често иде у комбинацији са технологијом која омогућава асинхроно учитавања података са сервера, без потребе за поновним учитавањем целе странице. Уколико заинтересованост и могућности ученика то дозвољавају, наставник може илустровати и овај принцип рада кроз неколико једноставних примера (на пример, функционалност аутоматског допуњавања поља за унос текста на основу скупа речи које се налазе у бази података).

Тема **Пројектни задатак** је посвећена пројектном раду ученика.

Први део пројектног задатка подразумева да тимови ученика креирају веб-странице која ће садржати одређене HTML елементе и имати дизајн описан CSS документом при чему садржај треба да представља истраживачки рад ученика, да буде аутентичан. Први део пројектног задатка може да буде или полазна основа за завршни пројектни задатак, или да се посматра као пројектни задатак мањег обима који служи за увежбавање и припрему за завршни пројекат. Наставник доноси одлуку о самом току и реализацији пројектних радова.

Други део пројектног задатка подразумева израду динамичког веб-сајта кроз коју би ученици сажели целокупно стечено знање веб-дизајна и веб-програмирања.

Пожељно је да, у виду пројекта на након обрађених тема Описни језик HTML и Стилски листови - језик CSS, ученици креирају веб-странице која ће садржати одређене HTML елементе при чему садржај треба да представља истраживачки рад ученика, да буде аутентичан.

Пример пројекта је да ученици у тиму креирају веб-сајт, који ће имати најмање пет повезаних страница, при чему странице треба да садрже следеће елементе:

* три наслова и поднаслова; текст мора бити подељен у пасусе при чему се могу користити визуелни елементи којим се пасуси јасније раздвајају (хоризонтална линија или подешавање доњих или горњих оквира пасуса);
* форматиран текст коришћењем CSS-а;
* најмање 5 текстуалних линкова, којим ће се линковани садржај отворити у потпуно новој страници;
* најмање 5 слика (користити слике преузете са интернета, водећи рачуна о ауторским правима);
* најмање 2 линка у виду слике, којим ће се линковани садржај отворити у потпуно новој страници;
* 2 табеле, бар у једној табели приказати оквир и спојене колоне или врсте;
* 3 угнежђена видео материјала који покривају тему коју су ученици изабрали;
* најмање по једну уређену и неуређену листу.

Цео веб-сајт треба буде стилизован коришћењем језика CSS при чему треба: користити стилизовање на нивоу веб-сајта, али и на нивоу појединачних страница; одредити фонт свим текстуалним деловима при чему кључне речи или реченице би требало да буду истакнуте неком опцијом за форматирање (искошена слова, подебљана, подвучена или промењена боја тог дела текста); прилагодити унутрашње и спољашње маргине неких елемената; променити боју текста и позадине и бар на једном елементу и поставити слику у позадини; увести бар једну класу елемената и стилизовати елементе на нивоу класе.

При реализацији првог дела пројектног задатка радити са најједноставнијим едитором, подстаћи ученике да примене стечена знања из рада са програмима за обраду текста и програмима за обраду слика и тако припреме садржај за креирање сопствене веб стране. Као пројектни рад ученици могу и да своје теме за матурски рад обраде коришћењем HTML и CSS језика и презентирају га у облику веб-странице.

Други део пројектног задатка подразумева израду динамичког веб-сајта кроз коју би ученици сажели целокупно стечено знање веб-дизајна и веб-програмирања.

Наставник предлаже теме за пројектне задатке. Неки могући примери:

* интернет продавница;
* формулар за регистрацију;
* веб форум.

При свему овоме неопходно је перманентно радити на развијању свести о важности поштовања правних и етичких норми при коришћењу интернета, критичком прихватању информација са веба, поштовању ауторских права при коришћењу информација са веба, поштовању права приватности.

### III. ПРАЋЕЊЕ И ВРЕДНОВАЊЕ НАСТАВЕ И УЧЕЊА

У процесу вредновања потребно је континуирано пратити рад ученика. У настави оријентисаној на достизање исхода вреднују се и процес и продукти учења. Прикупљање информација из различитих извора (свакодневна посматрања, активност на часу, учествовање у разговору и дискусији, самосталан рад, рад у групи, тестови) помаже наставнику да сагледа постигнућа (развој и напредовање) ученика и степен остварености исхода. Свака активност је добра прилика за процену напредовања и давање повратне информације. Важно је и ученике оспособљавати и охрабривати да процењују сопствени напредак у учењу.

У процесу праћења и вредновања значајну улогу имају домаћи задаци. Редовно задавање домаћих задатака (уз обавезну повремену проверу од стране наставника), помаже наставнику да стекне бољи увид у степен остварености исхода кроз анализу задатака које ученици нису умели да реше. Важно је и мотивисати ученике који редовно раде домаће задатке тако што ће њихов рад бити оцењен.

Вредновање активности у оквиру тимског рада се може обавити са групом тако да се од сваког члана тражи објашњење елемената урађеног рада и мишљење о сопственом раду унутар тима. Препоручује се да наставник са ученицима договори показатеље на основу којих сви могу да прате напредак у учењу, ученици се уче да размишљају о квалитету свог рада и о томе шта треба да предузму да би свој рад унапредили. Оцењивање тако постаје инструмент за напредовање у учењу. На основу резултата праћења и вредновања, заједно са ученицима треба планирати процес учења и бирати погодне стратегије учења.

Препоручено је да коначна оцена за сваког ученика буде добијена комбиновањем различитих начина оцењивања:

· активност на часу, учествовање у разговору и дискусији;

· редовна израда домаћих задатака;

· тестови - провера знања;

· пројектни рад, и појединачни и тимски.

Комбиновање различитих начина оцењивања помаже да се сагледају слабе и јаке стране сваког ученика. Приликом сваког вредновања постигнућа потребно је ученику дати повратну информацију која помаже да разуме грешке и побољша свој резултат и учење. Потребно је да наставник резултате вредновања постигнућа својих ученика континуирано анализира и користи тако да промени део своје наставне праксе.