Bosna i Hercegovina
Federacija Bosne i Hercegovine
KANTON SARAJEVO
Ministarstvo za odgoj i obrazovanje

Босна и Херцеговина
Федерација Босне и Херцеговине
КАНТОН САРАЈЕВО
Министарство за одгој и образовање

Bosnia and Herzegovina
Federation of Bosnia and Herzegovina
CANTON SARAJEVO
Ministry for Education

INSTITUT ZA RAZVOJ PREDUNIVERZITETSKOG OBRAZOVANJA
KANTON SARAJEVO, BOSNA I HERCEGOVINA

ИНСТИТУТ ЗА РАЗВОЈ ПРЕДУНИВЕРЗИТЕТСКОГ ОБРАЗОВАЊА
КАНТОН САРАЈЕВО, БОСНА И ХЕРЦЕГОВИНА

PRE-UNIVERSITY EDUCATION
INSTITUTE OF SARAJEVO CANTON
BOSNIA AND HERZEGOVINA

SOFTVER INŽINJERING

Nastavni plan i program/predmetni kurikulum zasnovan na ishodima učenja

SADRŽAJ

Riječi i pojmovi koji imaju rodno značenje korišteni u ovom dokumentu odnose se jednako na muški i ženski rod, bez obzira jesu li korišteni u muškom ili ženskom rodu.

PK1 – Opis nastavnog predmeta

Nastavni predmet Softver inžinjering omogućava učenicima priliku da otkriju, razumiju i primijene cjelokupan proces nastanka softverskog rješenja. On spaja teorijska znanja iz informatike i programiranja s praktičnim iskustvima koja odražavaju stvarni rad u IT industriji. Učenici se, kroz rad na konkretnim zadacima i projektima, osposobljavaju da prepoznaju potrebe korisnika, osmisle odgovarajuće rješenje, izrade tehničku dokumentaciju, implementiraju aplikaciju i evaluiraju rezultate. Takav pristup ne razvija samo tehničke vještine, nego i kritičko mišljenje, kreativnost, odgovornost i spremnost za saradnju.

U središtu ovog nastavnog predmeta je ideja da tehnologija nije sama sebi svrha, već sredstvo koje učenici koriste da bi unaprijedili svakodnevni život i riješili konkretne probleme. Zato se u nastavi podjednako pažnje posvećuje razvoju funkcionalnih i estetskih aspekata softvera, poštivanju etičkih standarda i razumijevanju sigurnosnih zahtjeva. Učenici uče kako da od početne ideje dođu do gotovog proizvoda prateći jasne faze rada – od istraživanja tržišta i specifikacije potreba, preko projektovanja i programiranja, do testiranja i predstavljanja rješenja. Na tom putu koriste alate i metodologije savremene industrije, poput verzionisanja koda, agilnog razvoja i vizuelnog dizajna interfejsa. Takav proces rada podstiče razvoj više ključnih kompetencija. Jezičko – komunikacijska pismenost jača kroz pisanje dokumentacije i prezentaciju projekata, dok se matematička pismenost razvija kroz rad s algoritmima, modelima i logičkim strukturama. Informatička pismenost se nadograđuje na visokom nivou – učenici ovladavaju naprednim softverskim alatima i razumiju kako se tehnologija može koristiti u kreativne, inovativne i društveno korisne svrhe. Kompetencija u nauci i tehnologiji oblikuje se kroz razumijevanje programskih paradigmi, struktura podataka i metoda testiranja, dok kreativno-produktivna kompetencija dolazi do izražaja u generisanju novih ideja, povezivanju znanja iz različitih oblasti i stvaranju originalnih rješenja.

Nastavni predmet Softver inžinjering ima snažnu poveznicu s drugim nastavnim predmetima. Matematika se oslanja na logičko mišljenje i preciznost potrebnu za programiranje, jezici na vještinu jasnog i stručnog izražavanja, a društvene nauke na razumijevanje korisničkih potreba i šireg konteksta primjene tehnologije. Osim toga, prirodno integriše međupredmetne teme poput digitalne pismenosti, održivog razvoja, građanskog odgoja i poduzetništva, jer učenici uče da tehnologiju koriste odgovorno, s razumijevanjem društvenih posljedica i s ciljem stvaranja vrijednosti.

Metodološki, nastava se odvija kroz kombinaciju teorijskog učenja i praktične primjene. Projekti su glavni pokretač procesa gdje učenici rade samostalno i u timovima, istražuju, planiraju i donose odluke, dok nastavnik djeluje kao mentor koji usmjerava i pruža podršku. Proces rada uključuje stalnu razmjenu ideja, evaluaciju i unapređivanje rješenja, što razvija otvorenost za konstruktivne kritike i spremnost na prilagođavanje. Vrednovanje ne obuhvata samo konačni proizvod, nego i trud, saradnju, kreativnost i sposobnost samoprocjene. Na taj način učenici uče da je uspjeh rezultat kontinuiranog učenja i poboljšavanja, a ne jednokratnog napora.
Kroz Softver inžinjering učenici ne stiču samo stručna znanja i tehničke vještine. Oni razvijaju samostalnost, odgovornost, organizacijske sposobnosti i sposobnost dugoročnog planiranja. Uče kako prepoznati problem, sagledati ga iz više uglova, osmisliti rješenje i dovesti ga do realizacije, a sve to u saradnji s drugima i uz poštivanje dogovorenih rokova i standarda kvaliteta. Takva iskustva im pomažu ne samo u daljem školovanju, nego i u aktivnom i odgovornom djelovanju u savremenom digitalnom društvu.

PK2 – Ciljevi učenja i podučavanja nastavnog predmeta

U nastavni nastavnog predmeta Softver inžinjering teži se ostvarenju sljedećih ciljeva:

- Razvijanje analitičkih vještina u svim fazama razvoja aplikativnog softvera – od prepoznavanja potreba i definisanja funkcionalnih i nefunkcionalnih zahtjeva, preko modeliranja sistema i odabira odgovarajućih razvojnih alata i okruženja, do pisanja, kompajliranja, testiranja i održavanja verzija izvornog koda.
- Razvijanje kreativnosti i inovativnosti pri pronalaženju rješenja problema, te prilagođavanju i unapređivanju postojećih rješenja korištenjem novih pristupa i tehnika razvoja softvera.
- Primjena stečenih znanja o aspektima razvoja softvera, upravljanju softverom, pouzdanosti i sigurnosti sistema kroz razvoj aplikativnih rješenja konkretne namjene.
- Razumijevanje i korištenje pravila životnog ciklusa razvoja softvera, uključujući planiranje, analizu, dizajn, implementaciju i pružanje podrške.
- Osiguravanje sigurne i etičke upotrebe tehnologija u učenju, komunikaciji i društvenim odnosima, uz poštivanje pravila zaštite privatnosti i identiteta, te zakonskih i etičkih normi digitalnog društva.

PK 3 - Oblasna struktura predmetnog kurikuluma

Izučavanje nastavnog predmeta Softver inžinjering realizuje se kroz tri oblasti koje su kreirane u odnosu na sami razvoj softver inžinjeringa i koje prate savremene trendove razvoja informacionih tehnologija:

A. Aspekti razvoja softvera

B. Upravljanje softverom

C. Pouzdanost i sigurnost sistema

A Aspekti razvoja softvera

Oblast „Aspekti razvoja softvera“ pruža učenicima sveobuhvatan uvid u proces stvaranja softverskih rješenja, povezujući teoriju, praksu i savremene metode softverskog inženjerstva. Učenici se upoznaju s osnovama discipline, etičkim pitanjima i praktičnim primjerima kroz studije slučaja, razvijajući sposobnost kritičkog promišljanja o uticaju softverskih rješenja. Kroz proučavanje softverskih procesa i modela, uključujući vodopad, spiralni i agilni razvoj (Scrum), učenici uče planirati, organizovati i pratiti različite faze razvoja softvera. Poseban naglasak stavljen je na inženjering zahtjeva i modeliranje sistema putem UML dijagrama, što omogućava precizno definisanje potreba korisnika i vizualizaciju softverskih struktura. Arhitektonsko projektovanje i dizajn softvera uče učenike važnosti arhitektonskih obrazaca, objektno – orijentiranih principa i upravljanja konfiguracijom. Implementacija, primjena razvojnih okruženja i evolucija softvera omogućavaju učenicima sticanje praktičnih vještina u razvoju, testiranju i održavanju aplikativnog softvera.

Ova oblast razvija analitičko razmišljanje, kreativnost, timski rad i sposobnost rješavanja stvarnih problema, pripremajući učenike za dalji profesionalni razvoj i primjenu savremenih pristupa u informacionim tehnologijama.

B Upravljanje softverom

Oblast „Upravljanje softverom“ uvodi učenike u ključne principe menadžmenta razvoja softvera, naglašavajući da softverski inžinjering nije samo tehnički proces, već i proces kojim se upravlja u okviru vremenskih, budžetskih i organizacijskih ograničenja. Učenici razvijaju razumijevanje upravljanja projektima kroz identifikaciju rizika, planiranje aktivnosti i koordinaciju timskog rada, što omogućava predviđanje i rješavanje potencijalnih problema u procesu razvoja softvera. Planiranje projekta uključuje analizu troškova, procjenu resursa i primjenu modela poput COCOMO za algoritamsko modelovanje troškova, čime se razvijaju sposobnosti strateškog razmišljanja i donošenja informisanih odluka. Upravljanje kvalitetom usmjerava učenike na evaluaciju softverskih rješenja, primjenu modela kvaliteta i razvoj sposobnosti objektivne procjene i unapređenja proizvoda. Upravljanje konfiguracijom naglašava značaj verzioniranja, kontinuirane integracije, upravljanja promjenama i izdanjima, uključujući upotrebu modernih alata poput Gita, čime učenici stiču praktična znanja za timski razvoj i održavanje velikih softverskih sistema. Ova oblast razvija analitičke i organizacijske vještine, odgovoran pristup radu, sposobnost koordinacije u timu i razumijevanje poslovnih i tehničkih aspekata razvoja softvera, pripremajući učenike za složene profesionalne zadatke i savremene izazove u IT industriji.

C Pouzdanost i sigurnost sistema

Oblast „Pouzdanost i sigurnost sistema“ fokusira se na završne faze razvoja softvera, gdje kvaliteta, pouzdanost i sigurnost postaju ključni elementi za uspjeh projekta. Učenici uče da testiranje softvera predstavlja kontinuirani proces, koji počinje testiranjem pojedinačnih jedinica koda, a završava sveobuhvatnim testiranjem cijelog sistema i konačnog izdanja softvera. Posebna pažnja posvećuje se razvoju vođenom testom, pristupu koji osigurava da softverska rješenja već u fazi implementacije zadovoljavaju zahtjeve korisnika i agilne standarde. U okviru ove oblasti učenici razumiju da kvalitetna isporuka softvera ne podrazumijeva samo instalaciju programa, već i pažljivo planiranje, evaluaciju i zadovoljavanje očekivanja korisnika. Podučavaju se tehnikama ranog otkrivanja i otklanjanja grešaka, shvatajući da kasno otkrivanje problema značajno povećava troškove i ugrožava reputaciju proizvođača softvera. Održavanje softvera predstavlja kontinuiranu aktivnost, koja uključuje otklanjanje grešaka, nadogradnju funkcionalnosti i prilagodbu sistema promjenama u okruženju. Učenici razvijaju sposobnost prepoznavanja potencijalnih rizika, planiranja rješenja i primjene sigurnosnih mjera koje štite integritet i povjerljivost sistema. Kroz ovu oblast učenici stiču ključne kompetencije u osiguravanju pouzdanosti softverskih proizvoda, razumijevanju procesa isporuke i održavanja, te razvijaju profesionalni, etički i odgovoran pristup razvoju i upotrebi softverskih sistema.

PK4 – Odgojno-obrazovni ishodi

Srednje IV
Godine učenja i podučavanja predmeta: 1

A **A. Aspekti razvoja softvera**	
A.IV.1 Analizira ideju softverskog procesa - koherentnog skupa aktivnosti za proizvodnju softvera.	A.IV.1.1 Primjenjuje etička i profesionalna pitanja za softverske inžinjere. A.IV.1.2 Koristi koncepte i modele softverskih procesa. A.IV.1.3 Raščlanjuje aktivnosti razvoja softverskih zahtjeva, implementacije, testiranja i evolucije. A.IV.1.4 Povezuje organizovanost procesa sa dizajnom i analizom softverskog rješenja. A.IV.1.5 Analizira faktore koji utiču na kvalitet i poboljšanje softverskog procesa.
A.IV.2 Razvija softversko rješenje primjenjujući agilne metode.	A.IV.2.1 Definiše agilne metode razvoja softvera i razlike u odnosu na planirani razvoj. A.IV.2.2 Koristi prakse agilnog razvoja kao što su refaktorisanje, programiranje u paru i razvoj prvog testa. A.IV.2.3 Primjenjuje Scrum metodologiju u projektnim zadacima. A.IV.2.4 Kombinuje agilne i planirane metode razvoja kod većih softverskih sistema.
A.IV.3 Dizajnira sisteme procesa kreirajući softverske zahtjeve i modelirajući dijagrame sistema.	A.IV.3.1 Razlikuje korisničke i sistemske zahtjeve u projektovanju softverskog rješenja. A.IV.3.2 Koristi grafičke modele za predstavljanje softverskih sistema. A.IV.3.3 Modelira dijagrame u UML-u (klase, sekvence, stanja, slučajevi upotrebe).
A.IV.4 Razvija koncepte arhitekture softvera i arhitektonskog projektovanja uvođenjem objektno orijentisanog softvera.	A.IV.4.1 Primjenjuje arhitektonske obrasce u dizajnu softverskih sistema. A.IV.4.2 Primjenjuje objektno - orijentisani proces projektovanja i dizajna. A.IV.4.3 Analizira ideje dizajnerskih obrazaca i načine ponovne upotrebe softverskog znanja. A.IV.4.4 Izdvaja ključna pitanja implementacije softvera, uključujući ponovno korištenje softvera i razvoj otvorenog koda.

KLJUČNI SADRŽAJI

Profesionalni razvoj softvera; Etika softverskog inžinjeringa; Studija slučaja; Modeli softverskih procesa; Agilne metode; Funkcionalni i nefunkcionalni zahtjevi; Kontekstni modeli; Modeli interakcije; Strukturni modeli; Modeli ponašanja; Inžinjering vođen modelom Odluke o arhitektonskom dizajnu; Objektno orijentisani dizajn koristeći UML; Dizajnerski obrasci; Pitanja implementacije; Razvoj otvorenog koda Evolucijski procesi; Naslijeđeni sistemi; Održavanje softvera

PREPORUKE ZA OSTVARIVANJE ISHODA UČENJA

1. Mogućnosti efikasnog učenja i podučavanja tematske cjeline- metodičke smjernice

Prijedlog rada na projektu za ostvarivanje ishoda učenja i indikatora:

1. Izabrati jednu od tradicionalnih metoda modelovanja koja će biti primijenjena u procesu razvoja konkretnog softvera. Preporučuju se kaskadni ili V model. U skladu sa izabranom metodom, predložiti detaljan plan projekta u kome treba:

- definisati kritične tačke (milestones) koje pokazuju šta treba da bude urađeno na projektu i do kada;
- navesti uloge svih članova projektog tima uz precizan opis šta ko od njih treba da uradi i do kada;
- navesti planirane rezultate za svaku kritičnu tačku; rezultati mogu biti softverski moduli, modeli, dokumentacija, kratka korisnička uputstva i sl.

2. Izraditi specifikaciju softverskih zahtjeva. Posebnu pažnju posvetiti:

- funkcionalnim zahtjevima sistema;
- zahtjevima kojima se definišu veze sistema sa okruženjem zahtjevima po pitanju performansi koje sistem treba da ispuni.

Specifikacija može, prema potrebi, da sadrži tekstualne opise zahtjeva, UML dijagrame, snimke ekrana, prijedloge izvještaja, tabela i dr.

3. Isprojektovati sistem sa aspekta njegove arhitekture i programskog kôda.

Predstaviti arhitekturu sistema u vidu modularne hijerarhije iz koje se jasno vidi koje komponente postoje u sistemu i kako su one međusobno povezane. Pri tome koristiti odgovarajući stil projektovanja.
Izabrati programski jezik koji će biti korišten pri implementaciji i obrazložiti njegov izbor. Opisati strukture podataka i algoritme koji će biti primijenjeni.
Napraviti strukturu programskih modula, tj. datoteka pomoću kojih će sistem biti implementiran. Navesti potrebne procedure i funkcije u okviru svakog programskog modula.
Pri projektovanju intenzivno koristiti sve vrste UML dijagrama.

4. Na izabranom programskom jeziku napisati programski kôd koji realizuje sistem. Datoteke sa izvornim kôdom dopuniti odgovarajućom unutrašnjom dokumentacijom, tj. komentarima. Na jednoj strani, priložiti primjer dobro dokumentovanog programskog kôda (početak datoteke sa zaglavljem).

2. Mogućnosti ostvarivanja međupredmetne povezanosti – međupredmetne korelacije

Međupredmetna povezanost nastavnog predmeta Softver inžinjering ostvaruje se kroz integraciju sadržaja i vještina iz različitih obrazovnih područja, čime se učenicima omogućava holističko razumijevanje i primjena stečenih znanja u širem kontekstu. Gradivni elementi softverskog sistema direktno se nadovezuju na ranije usvojena znanja iz Matematičkih osnova računarskih nauka i Napredne primijenjene informatike, gdje učenici koriste mehanizme nadzora rada sistema, administracije i izvještavanja. Ova povezanost omogućava učenicima da razumiju kako matematičke i logičke strukture utiču na funkcionalnost i stabilnost softverskih

rješenja. Kroz komponente baza podataka i programiranja, učenici primjenjuju znanja razvijajući sposobnost implementacije osnovne funkcionalnosti softverskog sistema i pravilnog upravljanja podacima. Kroz aktivnosti izrade projekata, razvijaju se komunikacijske vještine, posebno kroz obradu teksta i prezentaciju rezultata, što omogućava integraciju jezika, u kontekstu istraživanja literature i korištenja interneta kao pouzdanog izvora informacija. Numerička obrada podataka povezuje se s matematičkim sadržajem, dok kreativno računarstvo omogućava primjenu koncepta softverskog inžinjeringa kroz praktične zadatke u okviru Fizike, Psihologije, Sociologije i ostalih prirodnih nauka. Učenici razvijaju vještine grafičke obrade podataka, analize i prezentacije rezultata, što doprinosi i njihovoj sposobnosti timskog rada i organizacije projekata.

3. Mogućnosti odgojnog djelovanja i razvoja ključnih kompetencija

Razvoj nastavnog predmeta Softver inžinjering u gimnazijskom okviru ne ograničava se samo na sticanje tehničkih znanja, već značajno doprinosi odgoju i razvoju ključnih kompetencija učenika. Kroz rad na projektima i praktičnim zadacima, učenici razvijaju jezičko – komunikacijsku kompetenciju, jer čitaju, analiziraju i interpretiraju informativne tekstove na maternjem i engleskom jeziku, prezentiraju rezultate svojih projekata i argumentovano komuniciraju ideje unutar tima. Ova kompetencija se dodatno osnažuje kroz pripremu dokumentacije, izražavanje misli u softverskim specifikacijama i vođenje tehničkih bilješki. Ovaj nastavni predmet također podržava matematičku pismenost, jer učenici primjenjuju matematičke modele i logičke strukture pri dizajnu algoritama, analizi podataka i modeliranju softverskih sistema. Ovaj proces razvija sposobnost univerzalnog prikazivanja i objašnjavanja složenih fenomena, što je ključ za razumijevanje struktura i funkcionalnosti softvera. Kroz implementaciju i testiranje softverskih rješenja razvija se kompetencija u nauci i tehnologiji: učenici razumiju principe i metodologije razvoja softvera, primjenjuju ih u praktičnom kontekstu, prepoznaju veze tehnologije s društvom, kulturom i okolišem, te razvijaju interes za naučna i tehnološka istraživanja. Doprinos informatičkoj pismenosti je u kritičkom korištenju informacijsko-komunikacijske tehnologije, prikupljaju i vrednovanju informacija, razvoju kreativnosti, inovativnosti i odgovornom učestvovanju u digitalnim mrežama. Socijalne i građanske kompetencije se razvijaju kroz timski rad, konstruktivnu komunikaciju i poštivanje različitih kulturnih i društvenih normi, dok samoinicijativa i poduzetnička kompetencija jačaju kroz upravljanje projektima, prepoznavanje vlastitih sposobnosti, preuzimanje odgovornosti i konstruktivnu saradnju u grupama.

B. Upravljanje softverom	
B.IV.1 Upravlja softverskim projektima kroz planiranje aktivnosti, identifikaciju rizika i vođenje timskih članova.	B.IV.1.1 Identifikuje elemente organizacije softverskog projekta i podjele zadataka u skladu s kompetencijama projekta. B.IV.1.2 Procjenjuje rizike softverskog projekta i faktore koji utiču na motivaciju članova tima. B.IV.1.3 Predstavlja ključna pitanja koja utiču na timski rad, sastav tima, organizacija i komunikaciju unutar tima.
B.IV.2 Planira softverski projekt uz procjenu troškova, raspored	B.IV.2.1 Klasifikuje osnove obračuna troškova softvera i faktore koji utiču na cijenu softverskog sistema. B.IV.2.2 Opisuje planiranje projekta i korištenje trakastih dijagrama za vizuelizaciju rasporeda aktivnosti.

aktivnosti i primjenu različitih metodologija.	B.IV.2.3 Demonstrira agilno planiranje projekta kroz praktičnu aktivnost "Igra planiranja". B.IV.2.4 Koristi tehniku procjene troškova u konkretnim projektima i simulacijama.
B.IV.3 Koristi tehnike procjene troškova za analizu i planiranje budžeta u konkretnim softverskim projektima i simulacijama.	B.IV.3.1 Primjenjuje tehnike procjene troškova u planiranim softverskim projektima. B.IV.3.2 Analizira faktore koji utiču na troškove softverskog sistema i njihov uticaj na budžet. B.IV.3.3 Primjenjuje mehanizme za osiguranje kvalitete softvera u praksi. B.IV.3.4 Objašnjava upravljanje kvalitetom u agilnim metodama kroz razvoj timske kulture. B.IV.3.5 Procjenjuje atribute kvalitete softvera, softverske analitike i ograničenja softverskog mjerenja.
B.IV.4 Analizira procese i alate za upravljanje konfiguracijom softvera.	B.IV. 4.1 Analizira funkcionalnosti koje je potrebno osigurati u sistemu kontrole verzija. B.IV.4.2 Objašnjava izazove pri izgradnji sistema i koristi prednosti kontinuirane integracije i automatizovane izgradnje sistema. B.IV.4.3 Primjenjuje metode upravljanja promjenama softvera u praktičnim situacijama.

KLJUČNI SADRŽAJI

Upravljanje projektima: rizikom i ljudima; Timski rad; Planiranje projekta: Cijena softvera; Razvoj zasnovan na planu, projektni planovi, proces planiranja; Agilno planiranje; Tehnike procjene; Upravljanje kvalitetom softvera; Softverski standardi; Pregled projekta, proces pregleda projekta; Upravljanje kvalitetom i agilni razvoj; Softversko mjerenje; Upravljanje konfiguracijom: Upravljanje verzijama; Izgradnja sistema; Upravljanje promjenama; Upravljanje izdanjima

PREPORUKE ZA OSTVARIVANJE ISHODA UČENJA

1. Mogućnosti efikasnog učenja i podučavanja tematske cjeline - metodičke smjernice

Upravljanje projektima ima za cilj uvođenje upravljanja softverskim projektima kroz dvije važne upravljačke aktivnosti, upravljanje rizicima i upravljanje ljudima. Kroz prvi ishod učenja učenici će se upoznati sa glavnim zadacima voditelja projekta; pojmom upravljanja rizicima i nekim od rizika koji se mogu pojaviti u softverskim projektima; razumjet će faktore koji utiču na motivaciju i šta oni mogu da znače za voditelje softverskih projekata; razumjet će ključna pitanja koja utiču na timski rad, kao što su sastav tima, organizacija i komunikacija.

Planiranje projekta ima za cilj uvesti planiranje projekta, raspored i procjenu troškova. Razumjet će šta je uključeno u planiranje projekta i koristit će trakaste dijagrame za predstavljanje rasporeda projekta; uvedeni su u agilno planiranje projekta na temelju “Igre planiranja”; razumjet će tehnike procjene troškova i kako se model COCOMO II može koristiti za procjenu troškova softvera.

Upravljanje kvalitetom ima za cilj uvesti pojam upravljanja kvalitete softvera i softversko mjerenje. Učenici će se upoznati sa procesom upravljanja kvalitetom i znat će zašto je planiranje kvalitete važno; bit će svjesni

važnosti standarda u procesu upravljanja kvalitetom i znat će kako se standardi koriste u osiguranju kvalitete; razumjet će kako se pregledi i inspekcije koriste kao mehanizam za osiguranje kvalitete softvera; razumjet će kako se upravljanje kvalitetom u agilnim metodama temelji na razvoju kvalitete timske kulture; razumjet će kako mjerenje može biti od pomoći u procjeni nekih atributa kvalitete softvera, pojam softverske analitike i ograničenja softverskog mjerenja.

Kroz adekvatne primjere iz stvarnog života učenici će razumjeti izazove izgradnje sistema i prednost kontinuirane integracije i izgradnje sistema; razumjet će zašto je upravljanje promjenama softvera važno i osnovne aktivnosti u procesu upravljanja promjenama; razumjet će osnove upravljanja izdanjima softvera i kako se razlikuje od upravljanja verzijama.

Upravljanje kvalitetom objašnjava osnove upravljanja kvalitetom softvera, kao što se praktikuje u velikim projektima. Upravljanje kvalitetom se bavi procesima i tehnikama za osiguranje i poboljšanje kvaliteta softvera. Govori se o važnosti standarda u upravljanju kvalitetom, tj. koristi preglede i inspekcije u procesu osiguranja kvalitete. Na kraju, govori se o mjerenju softvera i raspravlja se o prednostima i problemima u korištenju metrike i softverske analize podataka u upravljanju kvalitetom.

Konačno, upravljanje konfiguracijom govori o upravljanju konfiguracijom, kritičnom pitanju za sve velike sisteme. Međutim, potreba za upravljanjem konfiguracijom nije uvijek očigledna učenicima koji će se baviti samo ličnim razvojem softvera, pa će se opisati različiti aspekti ove teme, uključujući upravljanje verzijama, izgradnji sistema, upravljanju promjenama i upravljanju izdanjima. Objašnjava se zašto je važna kontinuirana integracija ili svakodnevna izgradnja sistema. Važnost npr. Git-a koji se sve više koristi za podršku softver inžinjeringu od strane distribuiranih timova za upravljanje verzijama distribuiranih sistema.

2. Mogućnosti ostvarivanja međupredmetne povezanosti – međupredmetne korelacije

Oblast Upravljanje softverskim projektima omogućava povezivanje sa različitim nastavnim predmetima kroz praktičnu primjenu znanja. Sa Matematičkim osnovama kompjuterskih nauka i Naprednom primijenjenom informatikom povezuje se kroz mehanizme nadzora rada i administracije sistema te izradu izvještaja. Sa Bazama podataka i Programiranjem ostvaruje se saradnja kroz razvoj baza podataka i implementaciju funkcionalnosti. Bosanski jezik i književnost, Hrvatski jezik i književnosti, Srpski jezik i književnost doprinosi razvoju komunikacijskih vještina i pisanju tehničke dokumentacije, dok Engleski jezik osnažuje korištenje stručne literature i online resursa. Matematika pruža alate za numeričku obradu podataka, a Fizika i druge prirodne nauke doprinose kreativnom računarstvu i modeliranju. Psihologija i Sociologija pomažu u razvoju timske saradnje, prezentacijskih vještina i grafičke obrade podataka.

3. Mogućnosti odgojnog djelovanja i razvoja ključnih kompetencija –kompetencijski pristup

Učenici kroz ovu oblast razvijaju širok spektar ključnih kompetencija, počevši od jezičko-komunikacijske sposobnosti koja im omogućava da čitaju, razumiju i kritički analiziraju informativne tekstove, te jasno i precizno izražavaju vlastite ideje. Matematička pismenost potiče ih da koriste univerzalne matematičke obrasce mišljenja i prikazivanja radi objašnjavanja i modeliranja stvarnih situacija, dok kompetencija u nauci i tehnologiji obuhvata razumijevanje i primjenu različitih prikaza matematičkih elemenata i fenomena, odabir optimalnih rješenja i prepoznavanje veze tehnologije s naučnim, društvenim, kulturnim i ekološkim aspektima života.

Informatička pismenost razvija se kroz kritičko korištenje digitalnih tehnologija u prikupljanju, obradi i razmjeni informacija, uz poštovanje etičkih načela i zaštitu privatnosti. Kroz timski rad i zajedničke projekte jačaju socijalne i građanske kompetencije, učeći kako konstruktivno komunicirati, poštovati kulturne razlike i doprinositi zajednici. Istovremeno, razvijaju samoinicijativu i poduzetnički duh kroz preuzimanje odgovornosti, upravljanje rizikom i prepoznavanje vlastitih potencijala. Posebna pažnja posvećuje se kreativno-produktivnim kompetencijama, gdje učenici uče analizirati i vrednovati informacije, razlikovati činjenice od mišljenja,

povezivati ideje te razvijati kreativnost, estetsku osjetljivost i otvorenost prema novim iskustvima i perspektivama.

C. POUZDANOST I SIGURNOST SISTEMA	
C.IV.1 Analizira proces testiranja softvera primjenom odgovarajućih metoda, alata i faza testiranja radi osiguranja pouzdanosti i sigurnosti sistema.	C.IV.1.1 Primjenjuje faze testiranja softvera, od testiranja tokom razvoja do testiranja prihvatanja, u skladu s definisanim standardima kvaliteta. C.IV.1.2 Odabire izbor test slučajeva usmjerenih na identifikaciju i otklanjanje grešaka u programu. C.IV.1.3 Kreira planove i scenarije testiranja prije pisanja koda te implementira automatsko pokretanje testova. C.IV.1.4 Klasifikuje vrste testiranja (testiranje komponenti, sistema i izdanja) u kontekstu osiguranja pouzdanosti i sigurnosti softvera.
C.IV.2 Planira isporuku i održavanje softverskog sistema radi očuvanja pouzdanosti i sigurnosti.	C.IV.2.1 Izrađuje tehničku i korisničku dokumentaciju za isporuku softvera. C.IV.2.2 Organizuje obuku korisnika za upotrebu sistema. C.IV.2.3 Povezuje proces održavanja softverskog sistema s fazama njegovog životnog ciklusa.

KLJUČNI SADRŽAJI

Testiranje softvera: Greške i otkazi, klasifikacija grešaka; Vrste testiranja, jedinično testiranje, integraciono testiranje, sistemsko testiranje; Proces testiranja, Isporuka i održavalje softvera: Isporuka sistema, obuka korisnika, dokumentacija; Održavanje sistema, vrste održavanja, problemi održavanja, troškovi održavanja;

PREPORUKE ZA OSTVARIVANJE ISHODA UČENJA

1. Mogućnosti efikasnog učenja i poučavanja tematske cjeline– metodičke smjernice

U ovoj oblasti učenici analiziraju proces testiranja softvera i planiraju isporuku i održavanje sistema radi očuvanja njegove pouzdanosti i sigurnosti. Testiranje softvera obuhvata faze od testiranja jedinica tokom razvoja sistema do testiranja izdanja softvera. Učenici prepoznaju i primjenjuju faze testiranja – od testiranja tokom razvoja do testiranja prihvatljivosti od strane korisnika sistema. Upoznaju se s tehnikama koje pomažu u odabiru test slučajeva usmjerenih na otkrivanje grešaka te razlikuju razvojno testiranje od korisničkog testiranja.

Preporučuje se obrada kroz konkretne primjere:

Jedinično testiranje: metod „crne kutije“ (podjela na klase ekvivalencije, analiza graničnih vrijednosti, uzročno-posljedični grafovi) i metod „bijele kutije“ (pokrivanje iskaza, pokrivanje odluka, pokrivanje uslova),

Integraciono testiranje: integracija po principu „velikog praska“, integracija od dna ka vrhu, integracija od vrha ka dnu, „sendvič“ integracija.

Nakon isporuke, životni vijek softvera se nastavlja sve dok je u upotrebi. Softver evoluira kroz otklanjanje grešaka, poboljšanja i modifikacije, što je predmet održavanja. Učenici izrađuju dokumentaciju, organizuju obuku korisnika te povezuju proces održavanja sa fazama životnog ciklusa softvera.

Unutar projektnog zadatka učenici trebaju:

- Opisati sve sprovedene testove u okviru jediničnog testiranja po modulima,
- Navesti korišteni princip integracionog testiranja i dati hijerarhiju redoslijeda testiranja modula,
- Nacrtati dijagram zavisnosti broja pronađenih grešaka po sedmicama,
- Predložiti dio uputstva za korištenje koji se odnosi na jednu odabranu funkcionalnost sistema.

2. Mogućnosti ostvarivanja međupredmetne povezanosti – međupredmetne korelacije

Oblast Pouzdanost i sigurnost softverskog sistema pruža široke mogućnosti za integraciju znanja i vještina iz drugih nastavnih područja. Sadržaji vezani za gradivne elemente softverskog sistema (mehanizmi nadzora rada sistema, administracija, izvještavanje, baze podataka, razvoj funkcionalnosti) omogućavaju povezivanje s nastavnim predmetima Matematičke osnove kompjuterskih nauka i Napredna primijenjena informatika, gdje učenici razvijaju razumijevanje tehničkih principa i arhitekture softverskih rješenja. Nastavni predmeti Baze podataka i Programiranje direktno se povezuju kroz izradu baza, implementaciju osnovnih funkcionalnosti i testiranje koda, čime se ostvaruje primjena teorijskih znanja u praktičnim projektima. U okviru Bosanskog jezika i književnosti, Hrvatskog jezika i književnosti, Srpskog jezika i književnosti te Engleskog jezika, učenici razvijaju komunikacijske vještine, vježbaju obradu teksta te uče kako koristiti stručnu literaturu i internet izvore. Matematika doprinosi razvoju sposobnosti za numeričku obradu podataka, dok se Fizika i druge prirodne nauke povezuju kroz kreativno računarstvo i simulacije.
Psihologija i Sociologija daju okvir za razumijevanje korisničkog iskustva, timskog rada i prezentacijskih vještina, uz podršku kreativne i grafičke obrade podataka.

3. Mogućnosti odgojnog djelovanja i razvoja ključnih kompetencija – kompetencijski pristup

U oblasti Pouzdanost i sigurnost softverskog sistema razvija se širok spektar kompetencija koje povezuju različita znanja i vještine. Učenici unapređuju jezičko – komunikacijsku kompetenciju kroz čitanje, razumijevanje i analizu informativnih i tehničkih tekstova, kao i kroz jasnu i preciznu usmenu i pisanu komunikaciju u profesionalnom kontekstu. Istovremeno, jačaju matematičku pismenost kroz korištenje matematičkih metoda i prikaza za tumačenje podataka i opisivanje stvarnih procesa u radu softverskih sistema. Razvija se i kompetencija u nauci i tehnologiji, jer učenici razumiju i primjenjuju principe rada tehnoloških sistema, povezuju ih s naučnim dostignućima, društvenim i kulturnim aspektima te razvijaju svijest o uticaju tehnologije na okoliš i održivi razvoj. Informatička pismenost se nadograđuje kroz kritičko korištenje IKT-a za pretraživanje, obradu, predstavljanje i razmjenu informacija, uz poštivanje etičkih standarda i privatnosti.
Učenici također jačaju socijalne i građanske kompetencije kroz konstruktivnu saradnju, poštovanje komunikacijskih normi, međukulturno razumijevanje i kvalitetnu interakciju u timskom radu. Kroz zadatke se potiče samoinicijativnost i poduzetništvo, posebno u planiranju i vođenju projekata, procjeni rizika i prepoznavanju vlastitih snaga i slabosti.
Na kraju, nastava podstiče kreativno – produktivne kompetencije, omogućavajući učenicima da analiziraju i sintetiziraju informacije, razvijaju originalna rješenja, izraze kreativne ideje i estetske vrijednosti te da otvoreno i kritički promišljaju različite perspektive, integrišu nova iskustva i oblikuju sopstvene stavove.

PK5 – Učenje i podučavanje

Nastava Softver inžinjeringa usmjerena je na razvijanje informatičke pismenosti, istraživačkog duha i kritičkog mišljenja kod učenika. Kroz proces učenja učenici se osposobljavaju da istražuju, rasuđuju, donose zaključke, kreiraju samostalna rješenja i iznalaze nove puteve u naučnom djelovanju, ne samo unutar oblasti softver inžinjeringa nego i u multidisciplinarnom povezivanju s drugim naukama. Posebna pažnja posvećuje se razvoju vještina interpretacije podataka dobijenih u istraživačkim projektima, kao i sposobnosti evaluacije naučnih argumenata i dokaza. Temelj ovog učenja je sticanje i razvijanje znanja, vještina i stavova koji učeniku omogućavaju da uspješno ostvaruje vlastite interese, razvija lične potencijale i aktivno učestvuje u savremenom društvu. Posebno se naglašava razvoj tehničke i informatičke pismenosti potrebne za život i rad u složenom digitalnom okruženju. Učenici se podstiču da razumiju i primjenjuju informacione tehnologije za organizaciju i pristup podacima, komunikaciju, saradnju i sigurno korištenje digitalnih alata, uz isticanje etičke i odgovorne upotrebe tehnologije. U tom procesu razvijaju se radne navike, osjećaj odgovornosti, saradnja i timski rad, dok kreativnost i originalnost dolaze do izražaja u praktičnim zadacima i projektnim aktivnostima. Učenje softver inžinjeringa podrazumijeva istovremeno uključivanje kognitivnog, afektivnog i psihomotornog područja. Spoznaje utiču na stavove i ponašanja učenika, dok se kroz praktične aktivnosti razvija vještina djelovanja u svakodnevnom životu. Učenje se shvata kao aktivan proces u kojem učenik konstruiše znanje oslanjajući se na prethodna iskustva i postojeća saznanja. Aktivnim uključivanjem u izbor tema motivacija raste, a fleksibilnost nastavnog procesa daje nastavniku slobodu da osmisli metodički pristup i dinamiku učenja u skladu s mogućnostima i interesima učenika. Nastavnik je vodič u procesu sticanja znanja, pružajući smjernice kako ih razvijati i nadograđivati svakodnevno. Organizacija rada je raznovrsna i prilagođena potrebama i ciljevima nastave. Učenici mogu raditi individualno, u paru ili grupama, a grupni i projektni rad omogućava razvoj saradničkih kompetencija i dublje razumijevanje gradiva. Posebno je značajno projektno i problemsko učenje, integrisana nastava i simulacije kroz koje učenici stiču iskustvo koje podstiče praktičnu primjenu teorijskih znanja.

Nastava softver inžinjeringa se održava u kabinetima informatike, gdje su učenici podijeljeni u efikasne grupe koje mogu dati očekivane rezultate. U okviru svake grupe učenici se mogu podijeliti u skladu sa sklonostima učenika i nastavnikovoj procjeni usvojenosti znanja i razvijenosti vještina, a u skladu sa načelima izbornosti i inkluzije. Podjelu u manje grupe je moguće primjeniti u projektnom radu, problemskoj i integrisanoj nastavi, timskom radu, te tokom simulacija. Poželjno je da svaki učenik ima pristup vlastitom elektronskom uređaju kako bi nastava bila efikasnija i prilagođena zahtjevima savremenog okruženja. Učenici se kroz nastavu podstiču da razvijaju vještine samoregulacije – planiranja, organizacije i evaluacije vlastitog rada. Aktivno učenje prolazi kroz faze pripreme, realizacije i evaluacije. U pripremnoj fazi učenici analiziraju i postavljaju ciljeve, u fazi realizacije biraju strategije i prate njihovu učinkovitost, dok u evaluaciji vrednuju uspješnost svojih rješenja i donose odluke o daljem radu. Na taj način uče kako da samostalno preuzmu odgovornost za proces i rezultate učenja. Socijalna dimenzija učenja posebno je naglašena, budući da je učenje socijalna kategorija, i ona se ostvaruje kroz interakciju s vršnjacima i nastavnicima. Vršnjačka saradnja zauzima ključno mjesto, jer razvija osjećaj zajedništva, međusobnog uvažavanja i spremnosti na dijalog. Novi oblici komunikacije, uključujući digitalne alate i društvene mreže, čine sastavni dio svakodnevnice učenika i imaju važnu ulogu u razvoju saradničkih odnosa. Nastava Softver inžinjeringa uključuje teme sigurnosti na internetu, očuvanja privatnosti i odgovornog ponašanja u digitalnom prostoru. Učenici kroz zajedničke projekte uče kako da funkcionišu u grupi, razvijaju toleranciju i konstruktivnu komunikaciju, te koriste tehnologiju kao sredstvo povezivanja i građenja socijalnih vještina.

Poseban značaj pridaje se inkluzivnosti. Nastava Softver inžinjeringa treba da bude otvorena i dostupna svakom učeniku, bez obzira na njegove sposobnosti i teškoće. Inkluzija se ostvaruje kroz prepoznavanje i uvažavanje različitosti, korištenje tehnologije i specijalizovanih aplikacija koje olakšavaju pristup sadržajima, kao i kroz online platforme koje omogućavaju da učenici prate nastavu čak i u slučaju odsustvovanja. Time se sprječava isključivanje, a razvija se osjećaj pripadnosti i zdrava socijalna klima u odjeljenju.

Važno je naglasiti da se nastava Softver inžinjeringa povezuje s različitim nastavnim predmetima i naučnim disciplinama te je i planiranje nastave moguće u korelaciji sa prirodnim i društvenim naukama, ali i kroz jezičke i komunikacijske nastavne predmete. Informacijska tehnologija postaje spona koja omogućava integraciju učenja, potiče motivaciju i logičko mišljenje, jača sposobnost izražavanja i doprinosi uspjehu učenika u školovanju. Ova interdisciplinarna povezanost osigurava da Softver inžinjering ne ostane izolovan nastavni predmet, nego aktivno doprinosi razvoju ključnih kompetencija učenika u različitim područjima.

PK6 – Vrednovanje u predmetnom kurikulumu

Vrednovanje u nastavnom predmetu Softver inžinjering predstavlja kontinuiran i razvojno usmjeren proces koji omogućava praćenje, podržavanje i unapređivanje ostvarenja odgojno-obrazovnih ishoda učenja i postavljenih ciljeva. U središtu tog procesa stoji učenik, njegova aktivna uloga u sticanju znanja, razvijanju vještina i formiranju kompetencija koje su potrebne za razumijevanje, kreiranje i primjenu softverskih rješenja u različitim kontekstima. Vrednovanje u ovom nastavnom predmetu ne podrazumijeva samo mjerenje stepena usvojenog činjeničnog znanja, već i procjenu sposobnosti da se ono upotrijebi u praktičnim situacijama, kao i razvoj kritičkog mišljenja, sposobnosti analize, rješavanja problema i stvaranja novih ideja. Ono se temelji na tri međusobno povezane funkcije: vrednovanju za učenje, vrednovanju kao učenje i vrednovanju naučenoga. Formativno vrednovanje ili vrednovanje za učenje, omogućava nastavniku i učeniku da prate napredak tokom procesa učenja, identifikuju izazove i pravovremeno usmjere dalji rad. Takav oblik vrednovanja doprinosi jačanju motivacije i gradi partnerski odnos između nastavnika, učenika i roditelja. Vrednovanje kao učenje stavlja naglasak na učeničku samostalnost, jer podstiče samoprocjenu i samoregulaciju učenja, ali i razvija osjećaj odgovornosti za sopstveni napredak. Ovdje kriteriji moraju biti jasni i dostupni učeniku, kako bi znao u kojem smjeru treba usmjeriti vlastiti rad i kako da prepozna kvalitet rada svojih vršnjaka. Sumativno vrednovanje, ili vrednovanje naučenoga, obuhvata procjenu postignuća u određenom vremenskom periodu i daje ukupnu sliku o nivou ostvarenosti ciljeva i ishoda učenja.

Principi vrednovanja u ovom nastavnom predmetu temelje se na objektivnosti, transparentnosti i raznovrsnosti. Poseban naglasak stavlja se na pravovremenu i konstruktivnu povratnu informaciju koja ne ukazuje samo na greške, već ističe i ono što je dobro urađeno i daje prijedloge za dalje unapređenje. Vrednovanje je prilagođeno taksonomskim nivoima učenja – od prepoznavanja i reprodukcije znanja, preko njihove praktične primjene, do osmišljavanja novih rješenja i projekata. Budući da Softver inžinjering objedinjuje teorijske osnove i praktičnu implementaciju, vrednovanje mora obuhvatiti i znanje i vještine, ali i sposobnost učenika da ih povezuju, nadograđuju i koriste u različitim situacijama.

Važan element ovog procesa jeste uključenost učenika. Učenici imaju pravo i obavezu da budu upoznati s kriterijima i metodama vrednovanja, te da učestvuju u vlastitoj samoprocjeni i procjeni rada svojih vršnjaka. Samoprocjena i vršnjačko vrednovanje posebno su dragocjeni u radu na projektima i timskim zadacima, jer razvijaju osjećaj zajedničke odgovornosti, kritičku distancu prema vlastitom radu i uvažavanje tuđeg truda. Transparentan sistem, uz mogućnost pravovremenog uvida u rezultate i jasno obrazložene kriterije, jača povjerenje učenika u proces ocjenjivanja i motiviše ih da stalno napreduju.

Elementi vrednovanja u Softver inžinjeringu nisu ograničeni samo na provjeru faktografskog znanja. Poseban značaj ima procjena sposobnosti da učenik primijeni teorijska znanja u praksi, da razvija logičko i algoritamsko mišljenje, da analizira i uočava uzročno – posljedične veze, da prepoznaje i rješava probleme, kao i da razvija inovativna rješenja i nove ideje. U tom smislu vrednovanje obuhvata i razvoj timskog rada, komunikacijskih i prezentacijskih vještina, posebno kroz projekte i praktične zadatke. Povratna informacija u takvim slučajevima treba da ukaže na prostor za napredovanje, da ohrabri učenika i usmjeri ga na dalje korake u učenju.

Za procjenu učeničkih postignuća koriste se različite tehnike i instrumenti, od usmenih i pisanih provjera znanja do praktičnih zadataka, izrade portfolija i projekata. Učenici vode repozitorij vlastitih radova kroz koji se može pratiti njihov napredak tokom školske godine, a projektni zadaci vrednuju se višedimenzionalno – od kvaliteta ideje i planiranja, preko prikupljanja i obrade podataka, do završne prezentacije i evaluacije postignutih rezultata. Sve češće se koriste i digitalne platforme koje omogućavaju online provjere i kombinaciju klasičnih i digitalnih oblika vrednovanja.

Profil i stručna sprema

- Diplomirani matematičar-informatičar
- Magistar softverskog inžinjerstva
- Magistar matematike, nastavnički smjer
- Magistar matematičkih nauka, smjer teorijska kompjuterska nauka
- Svršenici Prirodno-matematičkog fakulteta informatičkog i računarskog usmjerenja
- Diplomirani inžinjer informatike i računarstva
- Svršenici Elektrotehničkog fakulteta informatičkog i računarskog usmjerenja
- Diplomirani informatičar
- Diplomirani inžinjer kompjuterskih nauka
- Diplomirani inžinjer informacionih sistema
- Magistar informacionih sistema
- Magistar informacionih tehnologija
- Magistar računarstva i informatike
- Magistar matematike i informatike
- Magistar saobraćaja, smjer kompjutersko-informacione tehnologije
- Magistar-Diplomirani inžinjer računarstva i informatike
- Magistar elektrotehnike - diplomirani inžinjer elektrotehnike, Odsjek automatika i elektronika
- Magistar elektrotehnike - diplomirani inžinjer elektrotehnike, Odsjek računarstvo i informatika
- Magistar elektrotehnike - diplomirani inžinjer elektrotehnike, Odsjek telekomunikacije
- Magistar elektrotehnike - diplomirani inžinjer