ԲՈՎԱՆԴԱԿՈՒԹՅՈՒՆ

Ներածություն 5

Գլուխ1. Գրական ակնարկ 7

1.1 Նմանատիպ համակարգերի հետազոտություն 7

1.2 Առաջարկվող համակարգի առավելությունները 11

1.3 Համակարգի մշակման համար օգտագործվող գործիքակազմ 11

1.4 Խնդրի դրվածք 12

Գլուխ 2. Օգտագործվող գործիքամիջոցների նկարագրություն 13

2.1 Java 13

2.1.1 Java-ի օգտագործումը ծրագրում 16

2.2 SpringBoot Framework 17

2.3 Intellij IDEA ծրագրային միջավայր: 19

2.4 Spring Security 19

2.5 PostgreSQL 20

2.5.1 PostgreSQL-ի օգտագործումը ծրագրում 22

2.6 React.js Framework 23

2.7 JavaScript 25

2.7.1 JavaScript-ի օգտագործումը ծրագրում 27

2.8 Frontend և Backend փոխհամագործակցություն 28

2.9 Ալգորիթմներ և մեթոդներ 29

Գլուխ 3. Համակարգի նկարագրություն 32

3.1 Knowledge Hub թեստավորման համակարգ 32

3.2 Համակարգի օգտատերերը եւ իրենց հնարավորությունները 34

Գլուխ 4. Տնտեսագիտություն 44

Գլուխ 5. Կենսագործունեության անվտանգություն 54

Գլուխ 6. Բնապահպանություն 60

Եզրակացություն 67

Գրականության ցանկ 68

Ներածություն

Գիտելիքների գնահատման համակարգը կրթական գործընթացի անբաժանելի մասն է, որը թույլ է տալիս օբյեկտիվորեն չափել ուսանողների ուսուցման մակարդակը: Ժամանակակից կրթական տեխնոլոգիաները պահանջում են թեստավորման գործընթացների ավտոմատացում և արդյունքների վերլուծություն: Կրթության որակի բարձրացման համար կարևոր խնդիր է դառնում համակարգի մշակումը, որը թույլ է տալիս թեստավորել, պահպանել արդյունքները և այդ տվյալների հիման վրա կատարել գիտելիքների բարելավման վերաբերյալ առաջարկություններ: Այս փաստաթուղթը քննարկում է գիտելիքների գնահատման գործիքի մշակումը, օգտագործելով թեստավորումը և տվյալների վրա հիմնված ամփոփումը:

Թվային տեխնոլոգիաների զարգացման հետ մեկտեղ ուսումն ու գիտելիքների գնահատումը անցնում են առցանց ձևաչափի: Ավանդական թեստավորման մեթոդները դառնում են ավելի քիչ արդյունավետ, քանի որ դրանք չեն տրամադրում տվյալների ավտոմատացված վերլուծություն և անհատականացված առաջարկություններ ուսանողների համար: Գիտելիքի գնահատման թվային համակարգերի ներդրումը կարող է բարձրացնել ճշգրտությունը, արագացնել թեստավորման գործընթացը և նվազեցնել ուսուցիչների ծանրաբեռնվածությունը առցանց կրթության աճող պահանջարկի համատեքստում նման համակարգերի մշակումը դառնում է հատկապես արդիական:

Աշխատանքի նպատակն է ստեղծել մի համակարգ, որը կծառայի ուսուցիչներին՝ հնարավորություն տալով գնահատել և հետևել ուսանողների գիտելիքների մակարդակին թեստերի միջոցով: Այս համակարգը պետք է ապահովի նաև արդյունքների վերլուծություն և առաջարկությունների ստեղծում՝ նպատակ ունենալով օգնել ուսանողներին բարելավել իրենց գիտելիքները: Այս համակարգը կունենա մեծ նշանակություն հատկապես կրթության մեջ, որտեղ գնահատման ու ուսումնասիրության գործիքների ավտոմատացումը կարող է օժանդակել

ուսանողներին առավել արագ ու նպատակային կերպով բարելավելու իրենց գիտելիքները՝ դարձնելով ուսուցման գործընթացը ավելի արդար և արդյունավետ:

Ավտոմատացված թեստավորման համակարգի ներդրումը նոր հնարավորություններ է բացում ուսումնական հաստատությունների համար: Այն թույլ է տալիս արագ թեստավորել մեծ թվով ուսանողներ՝ ուսուցիչներին տրամադրելով արդյունքների վերաբերյալ մանրամասն հաշվետվություններ և ուսուցման մեթոդների կատարելագործման վերաբերյալ առաջարկություններ: Ուսանողների համար նման համակարգերը դառնում են հետադարձ կապի աղբյուր, որն օգնում է կենտրոնանալ թույլ կողմերի վրա և արդյունավետորեն բաշխել ձեր ջանքերը քննություններին նախապատրաստվելիս:

Մշակված համակարգը կարող է ներդրվել ուսումնական հաստատություններում (դպրոցներ, քոլեջներ, բուհեր), խորացված վերապատրաստման դասընթացներում, ինչպես նաև օգտագործել կորպորատիվ դասընթացներում: Այն հարմար է ինչպես գիտելիքները ուսումնական պլանում, այնպես էլ մասնագիտական հմտությունները ստուգելու համար:

Գլուխ1. Գրական ակնարկ

1.1 Նմանատիպ համակարգերի հետազոտություն

Ժամանակակից թվային տեխնոլոգիաների զարգացման պայմաններում գիտելիքների ստուգումն ու ամփոփումը դարձել են կրթական և մասնագիտական ոլորտների կարևոր մասերից մեկը: Այս նպատակով մշակվել են մի շարք հավելվածներ և համակարգեր, որոնք հնարավորություն են տալիս օգտատերերին թեստերի միջոցով գնահատել իրենց հմտություններն ու գիտելիքները: Նման համակարգերի հիմնական նպատակն է նպաստել ուսումնական գործընթացի արդյունավետության բարձրացմանը՝ ապահովելով ինտերակտիվ և հարմարավետ մոտեցումներ: Համակարգի ստեղծման համար կատարվել են նմանատիպ համակարգերի ուսումնասիրություններ:

Կրթական ոլորտում լայն տարածում են գտել **Edmodo**-ն **Moodle**-ը և **Google Classroom**-ը, որոնք ուսուցման արդյունքների գնահատման հանրահայտ գործիքներ են: Այս հավելվածները թույլ են տալիս ուսուցիչներին և ուսանողներին ստեղծել թեստեր, վերլուծել արդյունքները և ամփոփել դրանք:

Google Classroom. Google Classroom-ը լայնորեն օգտագործվող հարթակ է կրթական միջավայրերի կառավարման համար: Այն անխափան կերպով ինտեգրվում է Google Workspace գործիքներին, ինչպիսիք են Google Drive-ը, Docs-ը, Sheets-ը և Meet-ը, ինչը թույլ է տալիս արդյունավետ կառավարել հանձնարարությունները, գնահատումը և հաղորդակցությունը:

Google Classroom-ի հիմնական առանձնահատկությունները.

1. **Առաջադրանքներ և գնահատում**. Ուսուցիչները կարող են ստեղծել, տարածել և գնահատել առաջադրանքները անմիջապես հարթակի միջոցով:
2. **Համագործակցության գործիքներ**. Ուսանողները կարող են համագործակցել նախագծերի վրա՝ օգտագործելով Google Փաստաթղթերը, Թերթերը և Սլայդները՝ խթանելով թիմային աշխատանքն ու ստեղծագործությունը:

3. **Վիրտուալ դասեր**. Google Meet-ի հետ ինտեգրված, այն աջակցում է կենդանի դասերի ձայնագրման գործառույթներով, ինչը հնարավորություն է տալիս ասինխրոն ուսուցմանը:
4. **Էլեկտրոնային պորտֆոլիոներ**. Google Drive-ը կարող է ծառայել որպես ուսանողների աշխատանքի պահոց՝ թույլ տալով նրանց պահպանել թվային պորտֆոլիոներ:

Համեմատելով ստեղծվող հավելվածի հետ կարող ենք ասել, որ Google Classroom-ի թիրախային լսարանը Google Workspace-ի հասանելիություն ունեցող դպրոցները և հաստատություններն են: Ստեղծվող վեբ հավելվածը ուղղված է ուսուցիչներին և ուսանողներին, ովքեր կարիք ունեն գիտելիքի թեստավորման և գնահատման մասնագիտացված հարթակի՝ պոտենցիալ առաջարկելով ավելի շատ անհատականացում թեստ ստեղծելու համար:

Google Classroom-ն առաջարկում է հանձնարարությունների և համագործակցության գործիքների լայն փաթեթ: Շեշտը դրվում է կառուցվածքային թեստավորման և առաջընթացի հետևման վրա, որն ավելի նեղ է, բայց պոտենցիալ ավելի կենտրոնացված:

Edmodo. Edmodo-ն լայնորեն օգտագործվող հարթակ է, որն ուղղված է կրթական միջավայրերի կառավարմանը: Այն նախատեսված է ուսուցիչներին և ուսանողներին օգնելու համար կազմակերպել դասեր, հանձնարարություններ, գնահատումներ և հաղորդակցություն: Edmodo-ն թույլ է տալիս ուսուցիչներին հետևել ուսանողների առաջընթացին, իսկ ուսանողներին՝ մասնակցել դասարանային ակտիվություններին, ներկայացնել աշխատանքներ և ստանալ արձագանքներ:

Edmodo-ի հիմնական առանձնահատկությունները

- **Առաջադրանքներ և գնահատում**. Ուսուցիչները կարող են ստեղծել, տարածել և գնահատել առաջադրանքները անմիջապես հարթակի միջոցով:

- **Համագործակցության գործիքներ**. Ուսանողները կարող են համագործակցել նախագծերի վրա՝ օգտագործելով Edmodo-ի քննարկման գործիքները, ինչպես նաև կիսվել ուսումնական նյութերով:
- **Դասարանական հաղորդակցություն**. Ուսուցիչները և ուսանողները կարող են հաղորդակցվել հաղորդագրությունների, էլեկտրոնային փոստի և քննարկումների միջոցով:
- **Հետադարձ կապ**. Ուսուցիչները կարող են տրամադրել անհատական մեկնաբանություններ և գնահատականներ առաջադրանքների համար:

Edmodo-ն առաջարկում է հիմնական գործիքներ դասերի և գնահատումների կառավարման համար, բայց այն հիմնականում կենտրոնացած է ընդհանուր ուսումնական փորձի վրա, այն է՝ հաղորդակցություն և առաջադրանքների բաժանում:

Moodle.

Moodle-ը բաց կոդով ուսուցման կառավարման համակարգ (LMS) է, որը նախատեսված է ուսանողների, ուսուցիչների և կրթական հաստատությունների համար՝ անձնականացված ուսուցման միջավայրեր ստեղծելու նպատակով: Այն լայնորեն օգտագործվում է ինչպես K-12, այնպես էլ բարձրագույն կրթության հաստատություններում՝ դասընթացներ առաջարկելու, առաջադրանքներ կառավարման և համագործակցության խթանման համար:

Moodle-ի հիմնական առանձնահատկությունները.

1. **Դասընթացների ստեղծում և կառավարում**: Ուսուցիչները կարող են ստեղծել դասընթացներ, կառավարել բովանդակությունը և կազմակերպել գնահատումներ: Moodle-ը աջակցում է տարբեր տեսակի բովանդակությանը, այդ թվում՝ տեսանյութեր, թեստեր, ֆորումներ և առաջադրանքներ:
2. **Համագործակցային գործիքներ**: Moodle-ը ներառում է գործիքներ, և խմբային գործունեություններ, որոնք խթանում են ուսանողների համագործակցությունը և համադասարանական ուսուցումը:
3. **Թեստեր և գնահատումներ**: Ուսուցիչները կարող են ստեղծել թեստեր՝ տարբեր հարցի ձևերով, ինչպես օրինակ՝ բազմակի ընտրություն, ճիշտ/սխալ

հարցեր և կարճ պատասխաններ։ Հարթակը նաև աջակցում է ավտոմատ գնահատման և կարծիքների տրամադրման հնարավորությանը։

4. **Դիտարկումներ և հաշվետվություններ**: Moodle-ը տրամադրում է մանրամասն վերլուծություններ ուսանողների կատարողականի մասին, ինչը թույլ է տալիս ուսուցիչներին հետևել առաջընթացին և տրամադրել անհատականացված կարծիքներ։
5. **Կարգավորում և դիզայնի ճկունություն**: Քանի որ Moodle-ը բաց կոդով հարթակ է, այն կարող է հարմարեցվել հաստատությունների հատուկ կարիքներին՝ առաջարկելով ճկունություն դիզայնի և գործառույթների հարցում։
6. **Մուտք մոբայլի միջոցով**: Moodle-ը ունի մոբայլ հավելվածներ ինչպես ուսանողների, այնպես էլ ուսուցիչների համար, որոնք հնարավորություն են տալիս ուսուցման շարունակմանը և առաջադրանքների, գնահատումների մուտք գործել սմարթֆոններից։
7. **Ինտեգրում երրորդ կողմի գործիքների հետ**: Moodle-ը ինտեգրվում է տարբեր արտաքին գործիքների և համակարգերի հետ, ինչպես օրինակ՝ Google Drive, Zoom և այլ LMS լուծումներ, ընդլայնելով իր գործառույթները։

Թիրախային լսարանը: Moodle-ը նախատեսված է կրթական հաստատությունների համար և կարող է սպասարկել լայն շրջանակի օգտատերեր՝ սկսած հիմնական դպրոցներից մինչև համալսարաններ։ Ստեղծվող կայքը հատուկ ուղղված է անհատ ուսանողներին և ուսուցիչներին՝ կենտրոնանալով գիտելիքների թեստավորման վրա։ Ընդհանուր առմամբ, Moodle-ը համընդհանուր և լայնածավալ հարթակ է, որն առաջարկում է բազմաթիվ կրթական գործիքներ, իսկ այս կայքը ունի ավելի նեղ համառոտություն՝ կենտրոնանալով թեստավորման և հետադարձ կապի վրա՝ առաջարկելով ավելի մասնագիտացված հարթակ։

1.2 Առաջարկվող համակարգի առավելությունները

Առաջարկվող համակարգը ունի մի շարք առավելություններ:

1. **Օգտատերերի համար հարմարավետություն**
    - Ինտուիտիվ և հարմարվող ինտերֆեյս ուսանողների և ուսուցիչների համար:
    - Մուտքի անհատականացում՝ գրանցման և լիազորման համակարգերի միջոցով:
2. **Արդյունավետ թեստավորման կառավարում**
    - Ուսուցիչների համար թեստերի ստեղծման և խմբագրման պարզ մեխանիզմ:
    - Տարբեր հարցերի ձևաչափերի աջակցություն:
3. **Անհատականացված ուսուցում**
    - Վերլուծություն՝ ուսանողների ուժեղ և թույլ կողմերի բացահայտման համար:
4. **Տվյալների պաշտպանություն**
    - Օգտատերերի տվյալների անվտանգություն և գաղտնաբառերի կոդավորում:
    - Համակարգի կայուն և անխափան աշխատանք:

1.3 Համակարգի մշակման համար օգտագործվող գործիքակազմ

1. Backend (տրամաբանություն և տվյալների պահպանում)

Աշխատանքի մշակումն իրականացվել է Java ծրագրավորման լեզվի միջոցով, օգտագործելով Intellij IDEA ինտեգրված ծրագրային միջավայրը, ինչպես նաև օգտագործվել են մի շարք գրադարաններ՝ տվյալների մշակման և դրանց հետ աշխատելու համար, ինչպիսիք են՝ Spring Jpa, Spring Web, Spring Security, Json Web Token, Javax Validation, Mail Sender, Lombok, PostgreSQL.

2. Վեբ տարբերակի ներդրման տեխնոլոգիաներ՝ Frontend (օգտագործողի միջերես):

- React.js Framework:

1.4 Խնդրի դրվածք

Խնդիր է դրվել, ստեղծել համակարգ, որը կծառայի ինչպես ուսուցիչներին, այնպես էլ դասախոսներին՝ հնարավորություն տալով գնահատել և հետևել ուսանողների գիտելիքների մակարդակին թեստերի միջոցով: Մշակված համակարգը կարող է ներդրվել ուսումնական հաստատություններում (դպրոցներ, քոլեջներ, բուհեր), խորացված վերապատրաստման դասընթացներում, ինչպես նաև օգտագործվել՝ կորպորատիվ դասընթացներում: Մշակվող հավելվածը նախատեսված է առցանց թեստավորման համար՝ ապահովելով ուսանողների գիտելիքների ստուգման արդար և վերահսկելի միջավայր: Այն հնարավորություն է տալիս դասախոսներին ստեղծել ու անցկացնել թեստեր, իսկ ուսանողներին՝ մասնակցել դրանց:

1. Դասախոսները կարող են ստեղծել թեստեր, սահմանել դրանց պայմանները, անցկացնել և ստանալ ուսանողների արդյունքները:
2. Ուսանողները կարող են մուտք գործել համակարգ, անցնել թեստերը և ստանալ իրենց գնահատականները:
3. Հավելվածը կարող է կիրառվել համալսարաններում, կրթական հաստատություններում և այլ կազմակերպություններում՝ գիտելիքների ստուգման գործընթացի թվայնացման և վերահսկելիության բարձրացման նպատակով:

Գլուխ 2. Օգտագործվող գործիքամիջոցների նկարագրություն

2.1 Java

Java-ն լայնորեն օգտագործվող ծրագրավորման լեզու է վեբ հավելվածներ գրելու համար: Միլիոնավոր Java հավելվածներ դեռ օգտագործվում են այսօր: Java-ն բազմահարթակ, օբյեկտի վրա հիմնված և ցանցակենտրոն լեզու է, որն ինքնին կարող է օգտագործվել որպես հարթակ: Այն արագ, անվտանգ և հուսալի ծրագրավորման լեզու է ամեն ինչի համար՝ բջջային հավելվածներից և ձեռնարկությունների ծրագրերից մինչև մեծ տվյալների հավելվածներ և սերվերային տեխնոլոգիաներ:

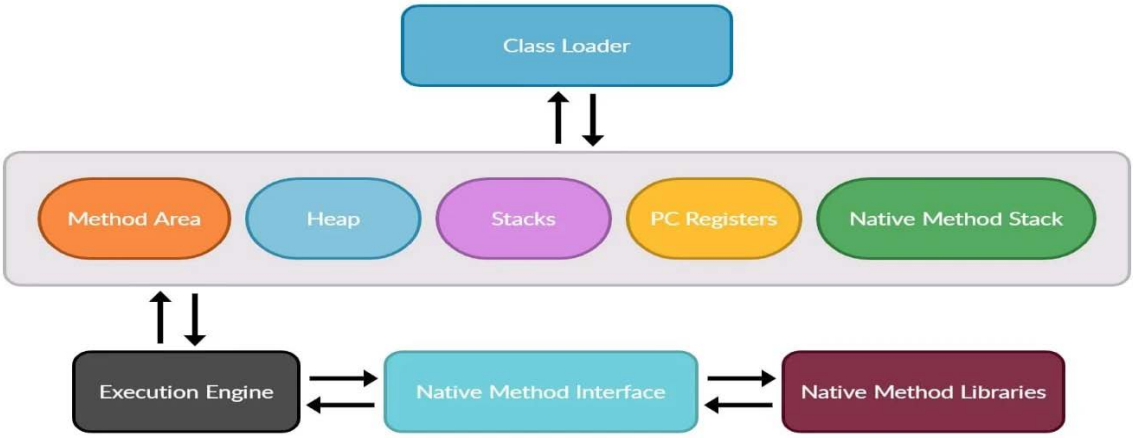
Java-ի պատմությունը որքան հետաքրքիր է, այնքան էլ ազդեցիկ: Ծրագրավորման լեզվի ճանապարհորդությունը սկսվել է 1991 թվականին, երբ Ջեյմս Գոսլինգը, Մայք Շերիդանը և Պատրիկ Նութոնը՝ Sun Microsystems-ի ինժեներների թիմը, որը հայտնի է որպես «Կանաչ թիմ», ձեռնամուխ եղան ստեղծելու նոր լեզու, որն ի սկզբանե կոչվում էր «Oak»: Oak-ը հետագայում վերանվանվեց Java՝ ոգեշնչված Java սուրճից և առաջին անգամ հրապարակվեց 1996 թվականին որպես Java 1.0: Այս նախնական տարբերակը տրամադրում էր առանց ծախսերի գործարկման միջավայր հանրաճանաչ հարթակներում՝ այն հասանելի դարձնելով լայն լսարանի համար:

Java-ն ժամանակի ընթացքում զարգացավ, Java 2-ը ներկայացրեց բազմաթիվ կոնֆիգուրացիաներ՝ հարմարեցված տարբեր հարթակների համար՝ ցուցադրելով իր բազմակողմանիությունը:

Java-ն նախագծվել է հիմնական սկզբունքներով՝ պարզություն, կայունություն, անվտանգություն, բարձր կատարողականություն, դյուրատարություն, բազմաշերտություն և դինամիկ մեկնաբանություն: Այս սկզբունքները Java-ին դարձրել են նախընտրելի լեզու տարբեր հավելվածների համար, ներառյալ շարժական սարքերը, ինտերնետ ծրագրավորումը, խաղերը և էլեկտրոնային բիզնեսը:

Java–ն օգտագործվում է դինամիկ պարունակությամբ վեբ էջեր, բազմատեսակ սպառողական ծրագրեր ստեղծելու, շարժական բազմապիսի էլեկտրոնային սարքերը ծրագրերով ապահովելու համար:

Java–ի տեղափոխելությունը պայմանավորված է նրանով, որ կոմպիլացվելուց հետո կոդն անմիջապես չի դառնում միջավայրին համապատասխան մեքենայական կոդ, այլ վերածվում է միջանկյալ կոդի, որը կոչվում է Java բայթկոդ: Վերջինս աշխատեցվում է Java վիրտուալ մեքենայի կողմից, որն արդեն հատուկ է տվյալ միջավայրին: Java վիրտուալ մեքենան գործում է որպես աբստրակցիայի լրացուցիչ շերտ Java պլատֆորմի և մեքենայի հիմքում ընկած սարքաշարի միջև: Java-ի սկզբնական կոդը կարող է գործարկվել միայն այն մեքենաների վրա, որոնցում տեղադրված է Java վիրտուալ մեքենա (JVM):



նկ. 2.1. Java վիրտուալ մեքենա (JVM)

Java-ի հիմնական առաձնահատկությունները

1. Անկախ հարթակ

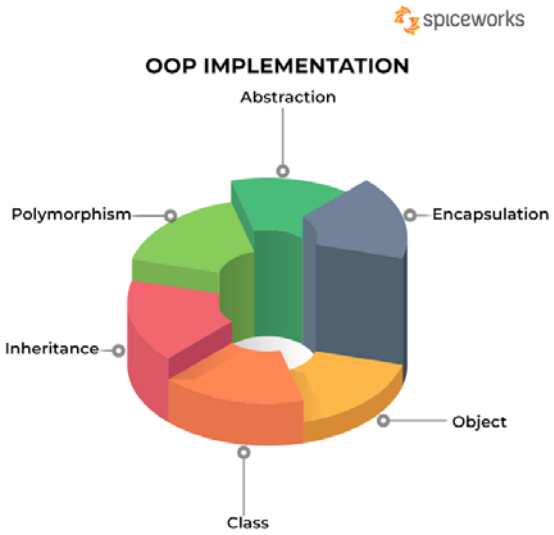
Կոմպիլյատորը սկզբնական կոդը փոխակերպում է բայթ կոդի, այնուհետև JVM-ն կատարում է կոմպիլյատորի կողմից ստեղծված բայթկոդը: Այս բայթ կոդը կարող է գործարկվել ցանկացած հարթակում՝ լինի դա Windows, Linux կամ macOS:

2. Օբյեկտ-կողմնորոշված ծրագրավորում

բյեկտ-կողմնորոշված ծրագրավորման չորս հիմնական հասկացություններն են.

- Աբստրակցիա
- Ինկապսուլյացիա
- Ժառանգություն
- Պոլիմորֆիզմ

Java-ն օբյեկտի վրա հիմնված լեզու է, որը նպաստում է օբյեկտների և դասերի օգտագործմանը: Ծրագրի կազմակերպումը օբյեկտների հավաքածուի պայմաններում օբյեկտի վրա հիմնված ծրագրավորման եղանակ է, որոնցից յուրաքանչյուրը ներկայացնում է դասի օրինակ:



նկ.2.2 Օբյեկտ-կողմնորոշված ծրագրավորում

3. Պարզություն :

Java-ի շարահյուսությունը պարզ և հեշտ է սովորել, հատկապես նրանց համար, ովքեր ծանոթ են C կամ C++-ին: Այն վերացնում է բարդ առանձնահատկությունները, ինչպիսիք են ցուցիչները և բազմաթիվ ժառանգականությունները, ինչը հեշտացնում է կոդ գրելը, վրիպազերծումը և պահպանումը:

4. Կայունություն

Java ծրագրավորման լեզուն հայտնի է իր բարձր կայունությամբ, ինչը այն դարձնում է հուսալի և լայնորեն օգտագործվող տարբեր ոլորտներում՝ սկսած վեբ ծրագրավորումից մինչև խոշոր ձեռնարկությունների համակարգեր: Java-ն նախագծված է այնպես, որ մեծ ուշադրություն է դարձնում ծրագրային սխալների կանխարգելմանը և դրանց հնարավորինս վաղ փուլում հայտնաբերմանը:

Լեզվի կառուցվածքը և թարմացվող տեխնոլոգիական հիմքը ապահովում են, որ հնարավոր սխալներն արդեն կոմպիլացիայի փուլում հայտնաբերվեն, ինչը նվազեցնում է կատարողականի ընթացքում առաջացող սխալների հավանականությունը: Բացի այդ, Java-ի կառավարվող հիշողության մոդելը (Managed Memory) և աղբյուրահավաքման մեխանիզմը (Garbage Collection) նպաստում են հիշողության արդյունավետ օգտագործմանը և թույլ չեն տալիս առաջացնել հիշողության արտահոսքեր կամ այլ լուրջ սխալներ:

2.1.1 Java-ի օգտագործումը ծրագրում

Java-ն օգտագործվել է ծրագրում որպես բեքենդի հիմք՝ ապահովելով ծրագրի տրամաբանությունը և տվյալների կառավարումը: Ծրագրում Java-ի կարևորության մի քանի ասպեկտները ներառում են՝

1. **Պլատֆորմից անկախություն**: Java-ի հիմնական առավելությունը բեքենդի զարգացման ժամանակ պլատֆորմից անկախությունն է: Սա նշանակում է, որ ծրագիրը կարող է աշխատել տարբեր օպերացիոն համակարգերում՝ Windows, Linux, macOS, ինչը շատ կարևոր է համացանցային հավելվածների համար:
2. **Օբյեկտ-կողմնորոշված ծրագրավորում**: Java-ն օբյեկտ-կողմնորոշված լեզու է, ինչը նպաստում է ծրագրի կառավարման պարզությանը և վերամշակելիությանը: Այսպիսով, իմ ծրագրում ստեղծվել են տարբեր օբյեկտներ՝ դասեր և ինստանսներ, որոնք իրականացնում են որոշակի տրամաբանություն, ինչպիսիք են օգտատերերի մենեջմենթը, ռոլերի համակարգը, օգտատերերի ավտորիզացիան և այլն:

3. **Spring Framework-ի ինտեգրում**: Java-ն օգտագործվել է Spring Boot Framework-ի հետ համատեղ, ինչը հեշտացրել է միկրոսերվիսների և API-ների ստեղծման գործընթացը: Spring Boot-ը ավտոմատ կերպով կառավարմամբ վերլուծում է առկա կախվածությունները և դինամիկ կերպով կատարում համապատասխան կոնֆիգուրացիաներ: Դա իդեալական էր բեքենդի արագ և արդյունավետ մշակման համար:
4. **PostgreSQL-ի հետ ինտեգրում**: Java-ն նաև օգտագործվել է PostgreSQL-ի հետ ինտեգրման համար՝ տվյալների բազայի մոդելի ստեղծման և կառավարման մեջ: Hibernate-ի միջոցով Spring Boot-ը կապում է Java-օբյեկտները PostgreSQL-ի տվյալների հետ՝ ապահովելով տվյալների պահպանում և մանիպուլյացիա:
5. **Spring Security**: Java-ն օգտագործվել է նաև անվտանգությունը ապահովելու համար՝ Spring Security-ի միջոցով, ինչը թույլ է տվել հաստատագրում (authentication) և թույլտվություն (authorization) կազմակերպել:

2.2 SpringBoot Framework

Spring Boot-ը Spring Framework-ի վրա հիմնված Java գրադարան է, որն օգտագործվում է Microservices-ների և Standalone վեբ հավելվածների արագ մշակման համար: Այն ապահովում է Spring Framework-ի ավտոմատ կոնֆիգուրացիա, ինչը թույլ է տալիս նվազեցնել ձեռքով կոնֆիգուրացիաների ծավալը և հեշտացնում է Spring հավելվածի մշակումն ու տեղակայումը:

Spring Boot-ի Հիմնական Առանձնահատկությունները

1. Ավտոմատ Կոնֆիգուրացիա (Auto-Configuration):

- Spring Boot-ը ավտոմատ կերպով կճանաչի dependency-ները և կկատարի համապատասխան Spring կոնֆիգուրացիաներ:
- Օրինակ, եթե դուք ավելացնում եք spring-boot-starter-web, ապա այն ավտոմատ կերպով կակտիվացնի Spring MVC-ի կոնֆիգուրացիան:

2. Starter Dependencies (Ստարտեր Գրադարաններ):

- Starter-ները (օրինակ՝ spring-boot-starter-web, spring-boot-starter-data-jpa) ներկայացնում են dependency-ների հավաքածուներ, որոնք անհրաժեշտ են որոշակի ֆունկցիոնալության համար:
- Օրինակ՝
    - spring-boot-starter-web-ը ներառում է Spring MVC-ի, Jackson-ի և այլ dependency-ներ:
    - spring-boot-starter-data-jpa-ն ներառում է Spring Data JPA և Hibernate:

3. Embeded Servers (Ներկառուցված Սերվերներ):

- Spring Boot-ը ներկառուցված սերվերներ է տրամադրում, ինչպիսիք են **Tomcat**, **Jetty**, և **Undertow**:
- Դա նշանակում է, որ դուք կարող եք ստեղծել և աշխատեցնել ձեր հավելվածը առանց առանձին սերվեր տեղադրելու:

4. Spring Boot CLI (Command-Line Interface):

- CLI-ն թույլ է տալիս գրել և աշխատեցնել Spring Boot հավելվածներ Groovy լեզվով՝ արագ փորձարկումների և մշակման համար:

5. Production-Ready Features (Արտադրական պատրաստականություն):

- Spring Boot-ը տրամադրում է գործիքներ, ինչպիսիք են **Actuator**-ը, որոնք թույլ են տալիս մոնիտորինգ և մետրիկաների (metrics) ապահովում:
- Actuator-ը տրամադրում է API-ներ, որոնք ցույց են տալիս հավելվածի վիճակը, health checks, request metrics և այլն:

6. Մինիմալ Կոնֆիգուրացիա:

- Spring Boot-ի շնորհիվ XML ֆայլերի և մեծածավալ Java կոնֆիգուրացիաների կարիք այլևս չկա:
- Application Properties/YAML: Հավելվածի կարգավորումները կարող են կատարվել application.properties կամ application.yml ֆայլերում:

7. Մոդուլայնություն:

- Spring Boot-ը լիովին ինտեգրվում է Spring-ի այլ մոդուլների հետ՝ թույլ տալով օգտագործել այնպիսի հնարավորություններ, ինչպիսիք են Spring Security, Spring Data, Spring Cloud և այլն:

8. Microservices Support:

- Spring Boot-ը հաճախ օգտագործվում է **միկրո ծառայությունների** մշակման համար, քանի որ ապահովում է.
    - Հեշտ ինտեգրում Spring Cloud-ի հետ
    - Նվազեցված կոնֆիգուրացիա
    - RESTful API-ների կառուցման գործիքները:

2.3 Intellij IDEA ծրագրային միջավայր:

IntelliJ IDEA-ն ծրագրավորման ինտեգրված զարգացման միջավայր (IDE) է, որն առաջարկում է գործիքակազմ ծրագրային ապահովման մշակման համար: Այն մշակված է JetBrains ընկերության կողմից և առանձնանում է իր լայն հնարավորություններով՝ աջակցելով տարբեր ծրագրավորման լեզուների, ֆրեյմվորքերի և գործիքների: IntelliJ IDEA-ն հատկապես ճանաչված է Java ծրագրավորման լեզվի համար նախատեսված իր բարձր ինտելեկտուալ վերլուծության և ավտոմատացման գործիքներով:

2.4 Spring Security

Spring Security - ն Java Spring Framework-ի ենթաբաղադրիչներից է, որը թույլ է տալիս հեշտությամբ ինտեգրել անվտանգության տարբեր մեխանիզմներ: Այն ապահովում է հաստատագրման (authentication), թույլտվության (authorization) ճկուն մեխանիզմներ, ինչպես նաև պաշտպանում է մի շարք հարձակումներից, ինչպիսիք ենք **CSRF**, **XSS** և այլ տիպային սպառնալիքներից՝ ապահովելով **OAuth2** և **JWT** ինտեգրացիան:

Հիմնական բաղադրիչները՝

Spring Security-ն կազմված է մի քանի հիմնական բաղադրիչներից, որոնք ներառում են՝

- Authentication Manager – կառավարում է հաստատագրման գործընթացը:
- Authorization Manager – պատասխանատու է օգտագործողի թույլտվությունների ստուգման համար:

- Security Context – պահում է տվյալ օգտագործողի անվտանգության կոնտեքստը:
- Filters – օգտագործվում են HTTP հարցումները վերլուծելու և ապահովագրելու համար:

Հաստատագրման մեխանիզմները՝

Spring Security-ն ապահովում է մի շարք հաստատագրման մեխանիզմներ, որոնք ներառում են՝

- Հիմնական հաստատագրում (Basic Authentication)
- Թոքենային հաստատագրում (JWT, OAuth2)
- LDAP և տվյալների բազայի հաստատագրում
- Միացման (SSO) տարբերակներ:

Թույլտվության կառավարում՝

Spring Security-ն ապահովում է թույլտվության ճկուն համակարգ, որը կարելի է կիրառել՝

- Մեթոդների մակարդակով (@PreAuthorize, @PostAuthorize անոտացիաներ)
- URL-երի մակարդակով (Spring Security configuration)
- Ռեսուրսների մակարդակով (RBAC, ABAC):

Spring Security-ի պաշտպանական հնարավորությունները՝

Spring Security-ն ապահովում է պաշտպանություն մի շարք սպառնալիքներից, այդ թվում՝

- CSRF (Cross-Site Request Forgery)
- XSS (Cross-Site Scripting)
- Session Fixation
- Clickjacking

2.5 PostgreSQL

PostgreSQL-ը հզոր, բաց կոդով օբյեկտ-հարաբերական տվյալների բազայի համակարգ է, որն օգտագործում և ընդլայնում է SQL լեզուն՝ համակցված բազմաթիվ գործառույթների հետ, որոնք ապահով կերպով պահում և չափում են տվյալների ամենաբարդ ծանրաբեռնվածությունը: PostgreSQL-ի ծագումը սկսվում է

1986 թվականին՝ որպես POSTGRES նախագծի մաս Բերկլիի Կալիֆորնիայի համալսարանի պրոֆեսոր Մայքլ Սթոունբրեյքերի ղեկավարությամբ և ունի ավելի քան 35 տարվա ակտիվ զարգացում հիմնական հարթակում: Նախագիծն ի սկզբանե անվանվել է POSTGRES՝ հղում անելով ավելի հին Էնգրես տվյալների բազային, որը նույնպես մշակվել է Բերքլիում: POSTGRES-ը նպատակ ուներ ավելացնել նվազագույն հատկանիշներ, որոնք անհրաժեշտ են տվյալների բազմաթիվ տեսակների ամբողջական աջակցության համար:

1996 թվականին նախագիծը վերանվանվեց PostgreSQL՝ ցույց տալու իր աջակցությունը SQL հարցումների լեզվին (չնայած PostgreSQL-ը դեռ սովորաբար կրճատվում է որպես Postgres): Աջակցողների նվիրված և բազմազան համայնքը՝ PostgreSQL Global Development Group-ը, շարունակում է կանոնավոր հիմնական և փոքր թողարկումներ կատարել անվճար և բաց կոդով տվյալների բազայի նախագծի վերաբերյալ:

PostgreSQL-ը մեծ համբավ է վաստակել իր ապացուցված ճարտարապետության, հուսալիության, տվյալների ամբողջականության, հզոր առանձնահատկությունների հավաքածուի, ընդարձակելիության և ծրագրաշարի հետևում բաց կոդով համայնքի նվիրվածության շնորհիվ՝ հետևողականորեն արդյունավետ և նորարար լուծումներ մատուցելու համար: PostgreSQL-ն աշխատում է բոլոր հիմնական օպերացիոն համակարգերի վրա, 2001թ.-ից ի վեր համապատասխանում է ACID-ին և ունի հզոր հավելումներ, ինչպիսիք են հայտնի PostGIS աշխարհատարածական տվյալների բազայի ընդլայնիչը: Զարմանալի չէ, որ PostgreSQL-ը դարձել է բաց կոդով հարաբերական տվյալների բազա շատ մարդկանց և կազմակերպությունների համար:

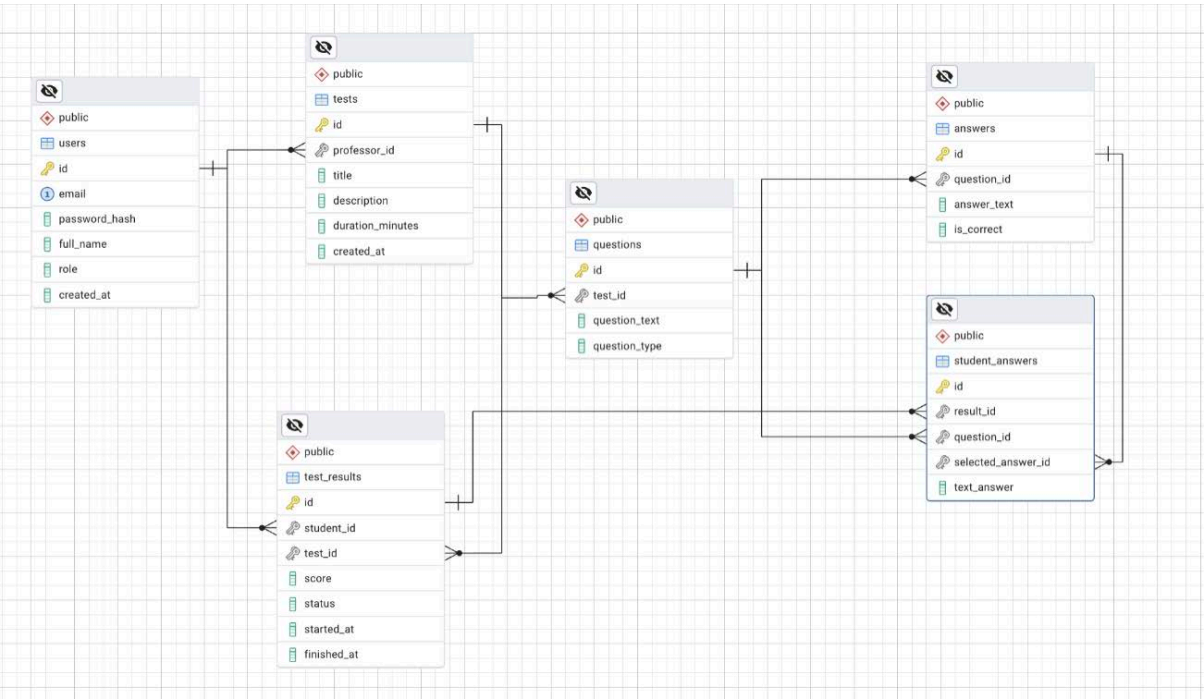
PostgreSQL-ն ունի բազմաթիվ առանձնահատկություններ, որոնք ուղղված են ծրագրավորողներին՝ ստեղծելու հավելվածներ, ադմինիստրատորներին՝ պաշտպանելու տվյալների ամբողջականությունը և ստեղծելու անսարքությունների հանդուրժող միջավայրեր, ինչպես նաև օգնելու ձեզ կառավարել ձեր տվյալները՝ անկախ նրանից, թե որքան մեծ կամ փոքր է տվյալների բազան: Բացի անվճար և բաց կոդով լինելուց, PostgreSQL-ը շատ ընդարձակելի է:

Օրինակ, հնարավորություն է տալիս սահմանել սեփական տվյալների տեսակները, ստեղծել հատուկ գործառույթներ, նույնիսկ գրել կոդ տարբեր ծրագրավորման լեզուներից՝ առանց տվյալների բազայի վերակոմպիլյացիայի:

PostgreSQL-ը փորձում է համապատասխանել SQL ստանդարտին, որտեղ նման համապատասխանությունը չի հակասում ավանդական հատկանիշներին կամ կարող է հանգեցնել ճարտարապետական սխալ որոշումների: SQL ստանդարտով պահանջվող գործառույթներից շատերը աջակցվում են, չնայած երբեմն մի փոքր տարբերվող շարահյուսություն կամ գործառույթ:

2.5.1 PostgreSQL-ի օգտագործումը ծրագրում

Տվյալ համակարգում տվյալների բազայում լինելու են հետևյալ աղյուսակները՝



նկ. 2.3. Տվյալների բազայի կառուցվածքը

- Օգտատերեր (users) աղյուսակը պահում է բոլոր օգտատերերի տվյալները՝ ինչպես ուսանողների, այնպես էլ դասախոսների և կապվում է test_results և tests աղյուսակների հետ:

- Թեստեր (tests) աղյուսակը պահում է բոլոր ստեղծված թեստերի տվյալները: Այն կապվում է questions աղյուսակի հետ՝ յուրաքանչյուր թեստը կարող է ունենալ բազմաթիվ հարցեր ինչպես նաև test_results աղյուսակի հետ՝ յուրաքանչյուր թեստի արդյունքները կապված են տվյալ թեստի հետ:
- Հարցեր (questions) աղյուսակը պահում է թեստերի հարցերը: Աղյուսակը կապվում է answers աղյուսակի հետ՝ յուրաքանչյուր հարցը կարող է ունենալ բազմաթիվ պատասխաններ: Ինչպես նաև կապվում է student_answers աղյուսակի հետ՝ յուրաքանչյուր հարցի պատասխանը կապված է ուսանողի կողմից տրված պատասխանին:
- Պատասխաններ (answers), աղյուակը պահում է հարցերի հնարավոր պատասխանները: Աղյուսակը կապվում է student_answers աղյուսակի հետ՝ յուրաքանչյուր պատասխանը կարող է կապված լինել ուսանողի տված պատասխանին:
- Թեստի արդյունքներ (test_results), աղյուսակը պահում է ուսանողների տրված թեստերի արդյունքները կապվում և է student_answers աղյուսակի հետ՝ այս աղյուսակում պահվում են ուսանողի պատասխանները:
- Ուսանողների պատասխաններ (student_answers) աղյուսակը պահում է ուսանողների տրված պատասխանները՝ տվյալ թեստի արդյունքների շրջանակում: Աղյուսակը կապվում է answers աղյուսակի հետ այսինքն ուսանողի պատասխանը կապված է որոշակի պատասխաններից և կապվում է questions աղյուսակի հետ՝ յուրաքանչյուր պատասխան կապված է որոշակի հարցի հետ:

2.6 React.js Framework

React.js-ը JavaScript – ի framework է, որը մշակվել է Facebook-ի (Meta) կողմից և նախատեսված է ինտերակտիվ օգտագործողի ինտերֆեյսների (UI) կառուցման համար: Այն հիմնված է կոմպոնենտային ճարտարապետության վրա և ապահովում է բարձր արդյունավետություն՝ օգտագործելով վիրտուալ DOM տեխնոլոգիան: React.js-ը լայնորեն կիրառվում է միաստոր էջային (SPA) և բարդ վեբ հավելվածների մշակման համար:

1. Կոմպոնենտային ճարտարապետություն (Component-Based Architecture)

- React.js-ը բաղկացած է վերաօգտագործվող կոմպոնենտներից, որոնք հանդիսանում են ծրագրի փոքր մոդուլներ:
- Կոմպոնենտները կարող են լինել դասային (Class Components) կամ ֆունկցիոնալ (Functional Components):
- Կոմպոնենտները ապահովում են մոդուլայնություն և հեշտացնում են նախագծի ընդլայնումը:

2. Վիրտուալ DOM (Virtual DOM)

- React-ը օգտագործում է Վիրտուալ DOM՝ արդյունավետության բարձրացման համար:
- Վիրտուալ DOM-ը ստեղծում է օբյեկտների վիրտուալ կոպիա, համեմատում այն նախորդ վիճակի հետ և թարմացնում միայն փոփոխված տարրերը:
- Այս մեթոդը նվազեցնում է բրաուզերի իրական DOM-ի թարմացումների քանակը, ինչը զգալիորեն բարձրացնում է հավելվածի արագագործությունը:

3. JSX (JavaScript XML) աջակցություն

- React-ը հնարավորություն է տալիս օգտագործել JSX՝ JavaScript-ի ընդլայնված շարահյուսություն, որը թույլ է տալիս HTML-ը գրել անմիջապես JavaScript կոդում:
- JSX-ը հեշտացնում է կոմպոնենտների ստեղծումը և ընթեռնելիությունը:

4. Unidirectional Data Flow (Մեկ ուղղությամբ տվյալների հոսք)

- React-ում տվյալների հոսքը միակողմանի է (one-way data binding), ինչը կանխում է անցանկալի փոփոխությունները:
- Տվյալների փոփոխությունները կատարվում են ծնող-երեխա (parent-child) հարաբերությամբ, ինչը ապահովում է ծրագրի կանխատեսելիությունը:

5. React Hooks (Քուկեր)՝ առանց վիճակի կառավարման դասերի

- React 16.8-ից հետո ներկայացվել են Hooks-ը, որոնք թույլ են տալիս օգտագործել վիճակ (state) և այլ React հնարավորություններ ֆունկցիոնալ կոմպոնենտներում՝ առանց դասերի:

6. Reusable Components (Վերաօգտագործվող կոմպոնենտներ)

- React-ի կոմպոնենտները հնարավոր է օգտագործել բազմակի՝ առանց կրկնօրինակելու կոդը:
- Կոմպոնենտները կարող են ունենալ սեփական տրամաբանությունը և ստանալ տվյալներ այլ կոմպոնենտներից:

7. Գրադարանների աջակցություն

- React-ը հեշտությամբ ինտեգրվում է այլ գործիքների և գրադարանների հետ, ինչպիսիք են՝
- State Management – Redux, Context API, MobX
- Routing – React Router
- UI Frameworks – Material-UI, Ant Design, Tailwind CSS
- API Requests – Axios, Fetch API

8. SSR (Server-Side Rendering) աջակցություն

- React-ը կարող է օգտագործվել սերվերային կողմում՝ Next.js-ի միջոցով, ինչը բարձրացնում է որոնման համակարգերի օպտիմիզացիան (SEO) և հավելվածի արագագործությունը:

9. React Native – Մոբայլ հավելվածների մշակում

- React-ի օգնությամբ հնարավոր է ստեղծել մոբայլ հավելվածներ՝ օգտագործելով React Native:
- React Native-ը թույլ է տալիս ստեղծել Android և iOS հավելվածներ նույն JavaScript կոդով:

10. Մինիմալ կոնֆիգուրացիա և արտադրական պատրաստականություն

- React-ը ապահովում է մաքուր և կազմակերպված նախագծային մոտեցում:
- CRA (Create React App) գործիքը թույլ է տալիս արագ ստեղծել նոր React հավելված՝ առանց բարդ կարգավորումների:
- React Developer Tools-ը թույլ է տալիս մոնիտորել և դեբագել React հավելվածները:

2.7 JavaScript

JavaScript-ը բարձր մակարդակի ծրագրավորման լեզու է, որը հիմնականում օգտագործվում է բրաուզերում ինտերակտիվությունն ու դինամիկություն

ապահովելու համար։ Այն թույլ է տալիս ծրագրավորողներին ստեղծել ինտերակտիվ վեբ էջեր՝ մանիպուլյացիա կատարելով էջերի բովանդակության, ձևերի, էլեմենտների և տվյալների հետ։ JavaScript-ը կիրառվում է տարբեր նպատակներով՝ սկսած տվյալների ստուգումներից մինչև վեբ հավելվածների ինտերակտիվության ապահովում։ Այն բաց կոդով լեզու է, և հաճախ համակցվում է HTML-ի ու CSS-ի հետ՝ ստեղծելու ամբողջական վեբ էջեր։

Ավելին, JavaScript-ը աջակցում է իրական ժամանակի ֆունկցիոնալությանը՝ օգտագործելով տեխնոլոգիաներ, ինչպիսիք են AJAX կամ WebSocket, ինչը թույլ է տալիս վեբ ծրագրերին գործել առանց ամբողջ էջի վերալիցքավորման։

1. Ինտերպրետացվող և բարձր մակարդակի լեզու

JavaScript-ը ինտերպրետացվող լեզու է, ինչը նշանակում է, որ այն չի պահանջում նախօրոք կոմպիլյացիա։ Ծրագրի կոդը անմիջապես վերլուծվում և կատարվում է վեբ բրաուզերի կողմից։ Սա ապահովում է արագ զարգացում և դյուրին փորձարկում։

2. Օբյեկտային-օրիենտացված (Object-Oriented) ծրագրավորում

JavaScript-ն աջակցում է օբյեկտների վրա հիմնված ծրագրավորմանը, ինչը թույլ է տալիս ստեղծել վերաօգտագործվող և մոդուլային կոդ։ Այն կիրառվում է ինչպես դասական օբյեկտային մոդելով, այնպես էլ պրոտոտիպային ժառանգմամբ։

3. Դինամիկ տիպավորում (Dynamically Typed)

JavaScript-ը թույլ է տալիս փոփոխականները սահմանել առանց տվյալների հստակ տիպը նշելու։ Տիպի որոշումը կատարվում է ծրագրի կատարման ընթացքում, ինչը հեշտացնում է կոդի գրառումը, սակայն կարող է հանգեցնել հնարավոր սխալների։

4. Ասինխրոն և իրադարձություններով կառավարվող (Asynchronous & Event-Driven)

JavaScript-ը հնարավորություն է տալիս աշխատել ասինխրոն կոդի հետ՝ օգտագործելով callback ֆունկցիաներ, Promises և async/await մեթոդները։ Սա կարևոր է վեբ հավելվածների համար, որտեղ անհրաժեշտ է միաժամանակ իրականացնել տարբեր գործողություններ՝ առանց օգտագործողի սպասեցնելու։

5. Պլատֆորմից անկախություն (Cross-Platform Compatibility)

JavaScript-ը կարող է աշխատել ցանկացած սարքի և օպերացիոն համակարգի վրա, քանի որ այն աջակցվում է բոլոր ժամանակակից վեբ բրաուզերների կողմից: Բացի այդ, Node.js-ի շնորհիվ այն դարձել է հզոր գործիք նաև սերվերային ծրագրավորման մեջ:

6. Մեծ գրադարանների և ֆրեյմվորքների աջակցություն

JavaScript - ն ունի գրադարաններ և framework - ներ, որոնք հեշտացնում են զարգացման գործընթացը.

- Frontend. React.js, Angular, Vue.js
- Backend. Node.js, Express.js
- Տվյալների վիզուալիզացում. D3.js, Chart.js
- Մեքենայական ուսուցում. TensorFlow.js

7. Մոդուլային և ճկուն կառուցվածք

Ժամանակակից JavaScript-ը հնարավորություն է տալիս օգտագործել մոդուլներ (ES6 modules), ինչը հեշտացնում է կոդի կազմակերպումը և ընթեռնելիությունը;

8. Բարձր արդյունավետություն և օպտիմիզացիա

Բրաուզերների Just-In-Time (JIT) կոմպիլյացիոն մեխանիզմը (օր.՝ Google V8) հնարավորություն է տալիս JavaScript - ին աշխատել առավել արդյունավետ և արագ, քան ավանդական ինտերպրետացվող լեզուները:

2.7.1 JavaScript-ի օգտագործումը ծրագրում

JavaScript-ը ծրագրում օգտագործվել է որպես հիմնական լեզու՝ առցանց ինտերակտիվություն ապահովելու համար: Այս լեզուն իր դերը խաղացել է ինչպես բրաուզերի կողմից կատարումներն արագացնելու, այնպես էլ օգտատերերի հետ ինտերակտիվ աշխատանքը կազմակերպելու համար: Ահա թե ինչպես է JavaScript-ը կիրառվել ծրագրում:

1. **Անհատական միջերեսի դինամիկություն**: JavaScript-ը հիմնականում օգտագործվել է բրաուզերում օգտագործողի գործողություններին արագ արձագանքելու համար: Սա ներառում է դինամիկ ձևերի ստուգումներ, թարմացվող տվյալներ, օգտատերերի մուտքի տվյալների ստուգումներ :
2. **Ազատ ինտերֆեյսի ստեղծում**: JavaScript-ը օգտագործվել է նաև տարբեր բջջային սարքերի և դեսքթոփ մոնիտորների համար դինամիկ ինտերֆեյսների ստեղծման համար՝ ապահովելով ճիշտ չափումներով և պրոֆեսիոնալ տեսք: Համոզվել, որ ծրագրի UI-ն ավտոմատ կերպով հարմարեցվում է տարբեր էկրանների մեծությունների համար՝ առանց հավելյալ բարդությունների:
3. **AJAX (Asynchronous JavaScript and XML)**: JavaScript-ը կիրառվել է նաև AJAX տեխնոլոգիայի միջոցով, որը թույլ է տալիս տվյալների փոխանցում առանց էջի ամբողջական վերալիցքավորման: Սա անհրաժեշտ էր օգտատերերի և բեռնման ինտերֆեյսի միջև տվյալների արդյունավետ փոխանցման համար՝ հատկապես տեսնելով պլատֆորմի ստատուսները, արդյունքները և նոր թեստերի թարմացումը:

2.8 Frontend և Backend փոխհամագործակցություն

1. Օգտատիրոջ գործողությունը ֆրոնտենդում: Օրինակ՝ օգտատերը լրացնում է մուտքագրման login և սեղմում է "Մուտք գործել" կոճակը:
2. JavaScript-ը ուղարկում է հարցում բեքենդին: Օրինակ՝ օգտագործելով fetch() կամ Axios, ֆրոնտենդը ուղարկում է POST հարցում՝ պարունակելով օգտատիրոջ մուտքանունը և գաղտնաբառը:
3. Բեքենդը մշակում է հարցումը:
   - Spring Boot Controller-ը ստանում է հարցումը և փոխանցում այն համապատասխան Service-ին:
   - Եթե մուտքի տվյալները ճիշտ են, բեքենդը վերադարձնում է JWT թոքեն, որը ֆրոնտենդը պահում է:
4. Ֆրոնտենդը պահում է թոքենը և կատարում պաշտպանված հարցումներ

- Օգտատերը հաջողությամբ մուտք է գործում, և JWT թոքենը պահվում է Local Storage-ում կամ Cookies-ում:
- Հաջորդ հարցումների ժամանակ ֆրոնտենդը փոխանցում է այդ թոքենը Authorization վերնագրի մեջ:

5. Բեքենդը ստուգում է թույլտվությունը
   - Spring Security-ն ստուգում է Authorization վերնագրով եկած JWT թոքենը:
   - Եթե թոքենը վավեր է, օգտատիրոջ տվյալները վերադարձվում են:
   - Եթե թոքենը վավեր չէ կամ ժամկետանց է, բեքենդը վերադարձնում է 401 Unauthorized:

2.9 Ալգորիթմներ և մեթոդներ

- Գործունեության հետևման ալգորիթմ:

Օգտագործում է JavaScript API, որը FOCUS_LOST իրադարձություն է ուղարկում backend:

JavaScript-ը հետևում է ուսանողի ակտիվությանը: Եթե ուսանողը հեռանում է թեստային պատուհանից և կենտրոնացած չէ տվյալ տարրի վրա, ապա ստեղծվում է FOCUS_LOST իրադարձություն: Այս իրադարձությունը ուղարկվում է բեքենդի, որտեղ հետագայում պրոցեսը որոշում է, թե՝ պետք է իրականացվի արգելափակում կամ թեստը ձախողվի: Բեքենդը ստանում է ուսանողի սխալ գործունեությունը նշող FOCUS_LOST իրադարձություն և եթե ուսանողը անցել է մեկ այլ պատուհանի, ապա թեստը ավարտվում է:

FOCUS_LOST իրադարձությունը հետևում ենք JavaScript-ի blur իրադարձությունով:

blur իրադարձություն – Երբ ուսանողը սեղմում է այլ պատուհան կամ քլիք է անում այլ ծրագրի վրա, ծրագրում ստեղծված sendFocusLostEvent() ֆունկցիան ուղարկում է հաղորդագրություն backend-ին՝ ուսանողի սխալ գործունեության մասին: Java backend-ը (Spring Boot Controller) ստանում է ուսանողի տվյալները և թեստը ավարտվում է առանց բազայում տվյալներ պահելու

- HMAC + SHA-256 ալգորիթմը և դրա օգտագործումը Spring Security –ում

Հաշվողական անվտանգության ապահովման համար շատ տարածված է Message Authentication Code (MAC) ալգորիթմների կիրառումը, որոնք ապահովում են հաղորդագրության ամբողջականությունն ու իսկությունը: Այդպիսի մեթոդներից է HMAC (Hash-based Message Authentication Code) ալգորիթմը, որն օգտագործում է հեշ ֆունկցիա և գաղտնի բանալի՝ տվյալների անվտանգությունը երաշխավորելու համար:

HMAC-ը MAC-ի մի տեսակ է, որը հիմնված է հեշ ֆունկցիայի վրա: Այն գործում է հետևյալ կերպ՝

1. Ընդունում է գաղտնի բանալի և հաղորդագրություն որպես մուտքային տվյալներ:
2. Կիրառում է տվյալների վրա SHA-256 հեշ ֆունկցիան՝ ապահովելով 256 - բիթանոց ելքային արժեք:
3. Վերադարձնում է վերջնական HMAC արժեքը, որը ապահովում է հաղորդագրության ամբողջականությունը:

HS256-ը HMAC-ի կոնկրետ իրականացումն է, որը հիմնված է SHA-256 հեշ ֆունկցիայի վրա և լայնորեն կիրառվում է JSON Web Token (JWT) ստուգման մեխանիզմներում:

HS256-ի Օգտագործումը Spring Security-ում՝

Spring Security-ն ապահովում է JWT վավերացման մեխանիզմ, որտեղ HS256-ը կիրառվում է որպես հիմնական ստորագրման մեթոդ: JWT-ն կազմված է երեք հիմնական բաղադրիչից՝

- Header – Նշում է ալգորիթմը (օրինակ՝ HS256):
- Payload – Պարունակում է օգտվողի մասին տեղեկատվություն:
- Signature (Ստորագրություն) – Գաղտնի բանալիով ստորագրված հատված, որը ստեղծվում է HMAC + SHA-256 մեթոդով:

Spring Security-ում JWT-ի վավերացումն իրականացվում է հետևյալ քայլերով՝

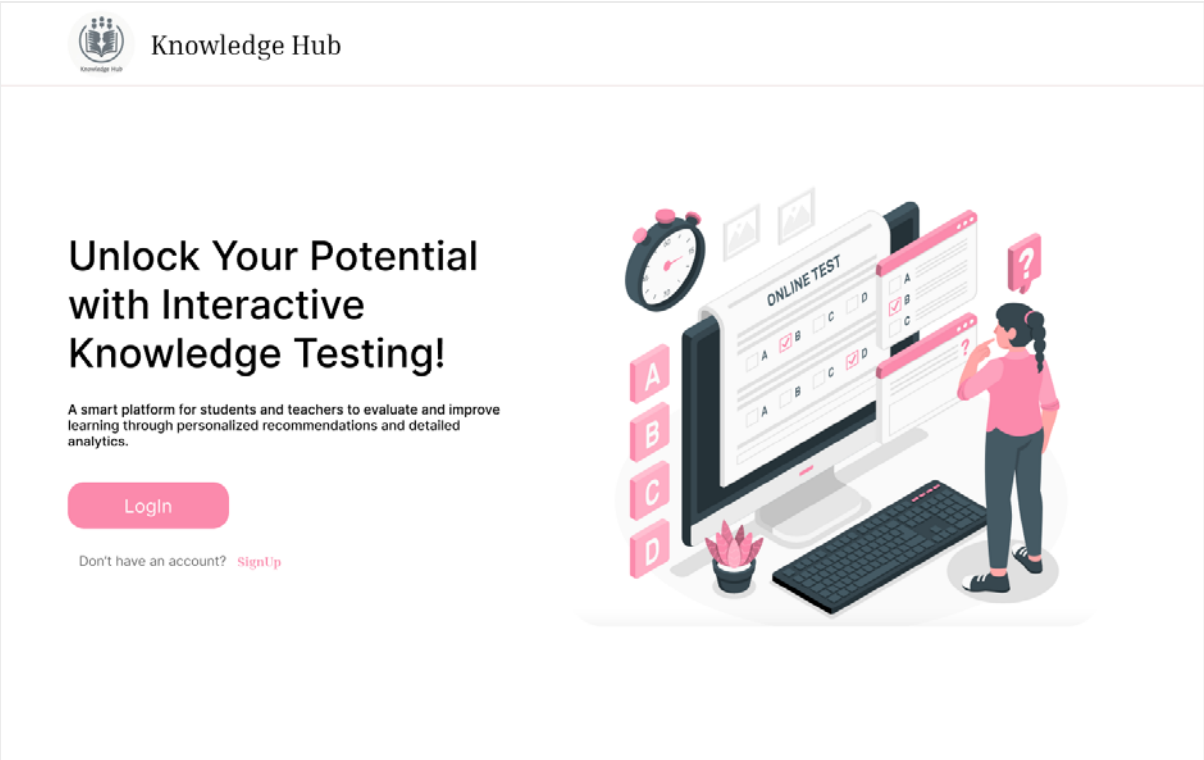
1. Օգտագործողը մուտք է գործում համակարգ՝ տրամադրելով նույնականացման տվյալներ:

2. Վավերացման մեխանիզմը (Spring Security) ստեղծում է JWT՝ օգտագործելով HS256 ալգորիթմը։
3. Յուրաքանչյուր հարցման ժամանակ JWT-ն ստուգվում է՝ համապատասխան գաղտնի բանալիի օգնությամբ։
4. Եթե ստորագրությունը վավեր է, ապա օգտվողին թույլատրվում է մուտք գործել ռեսուրս։

Գլուխ 3. Համակարգի նկարագրություն

Այս բաժնում կներկայացնենք մշակված համակարգի աշխատանքի գրաֆիկական նկարագրությունը՝ դասախոսների եւ սովորողների գրանցումը եւ մուտքը կայք: Ինչպես նաեւ՝ թեստային առաջադրանքների կազմումը, կառուցվածքը, իրենց իրականացման ձևով և վերջնական արդյունքների ամփոփմամբ ու ներկայացմամբ:

3.1 Knowledge Hub թեստավորման համակարգ

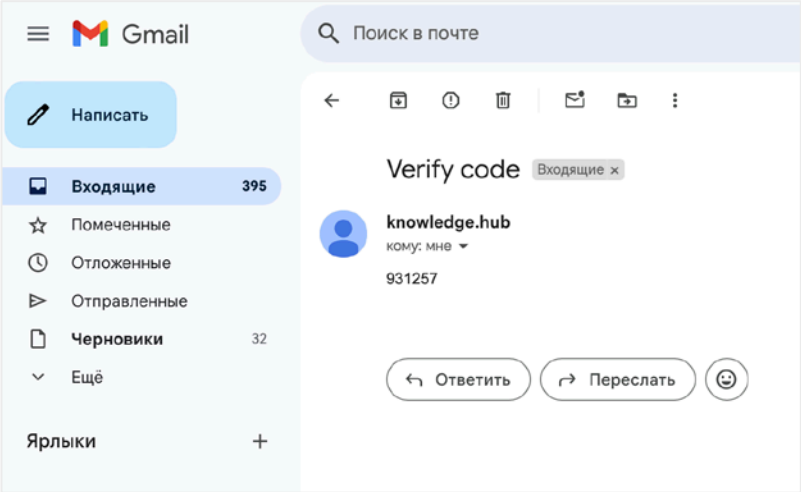


Նկ. 3.1. Knowledge Hub թեստավորման համակարգ, գլխավոր էջ

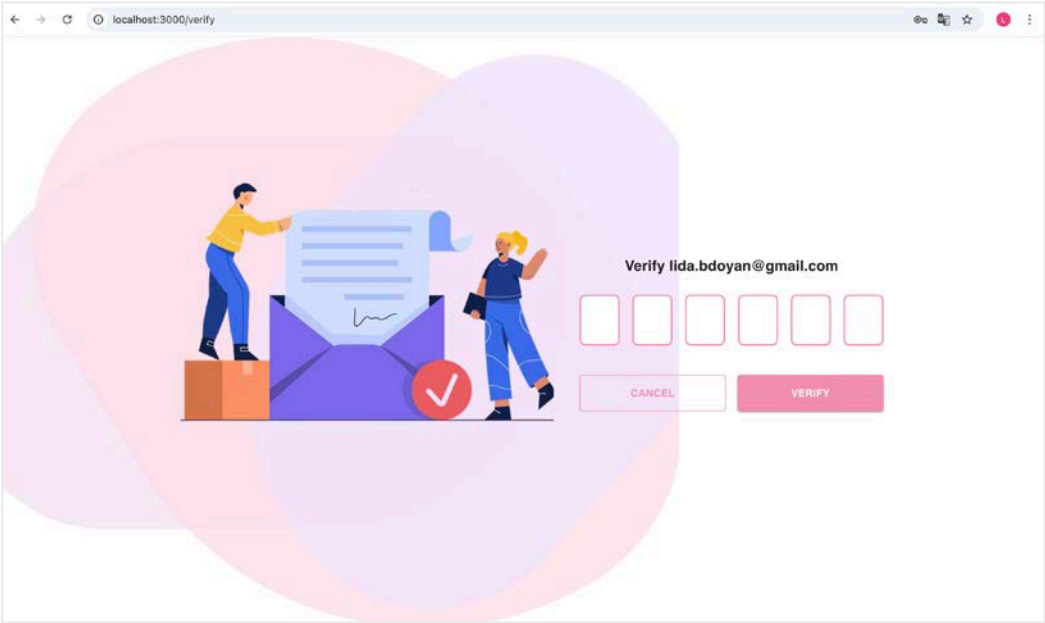
Աշխատանքում մշակված համակարգի գլխավոր էջը բաղկացած է երկու հիմնական բաժիններից՝ **Login** և **Sign Up**: Այս բաժինները նախատեսված են համապատասխանաբար ուսանողների և դասախոսների մուտքն ապահովելու համար: **Login** բաժնի միջոցով համակարգ մուտք կգործեն արդեն գրանցված օգտատերերը՝ մուտքագրելով իրենց էլեկտրոնային հասցեն և գաղտնաբառը: Իսկ

Sign Up բաժինը նախատեսված է նոր ուսանողների համար, ովքեր պետք է լրացնեն գրանցման ձևը՝ նշելով իրենց անձնական տվյալները, ինչպիսիք են անունը, ազգանունը, էլեկտրոնային հասցեն և ընտրած գաղտնաբառը:

Ուսանողները, գրանցման գործընթացը հաջողությամբ ավարտելուց հետո, իրենց տրամադրած էլեկտրոնային հասցեին կստանան **հաստատման (verification) կոդ**, որը մուտքագրելով կկարողանան ակտիվացնել իրենց հաշիվը և ստանալ մուտքի իրավունք կայք: Այս գործընթացը ապահովում է գրանցման անվտանգությունը և թույլ է տալիս բացառել կեղծ օգտահաշիվների ստեղծումը:



Նկ. 3.2. Mail-ին ուղարկվող վավերացման կոդ

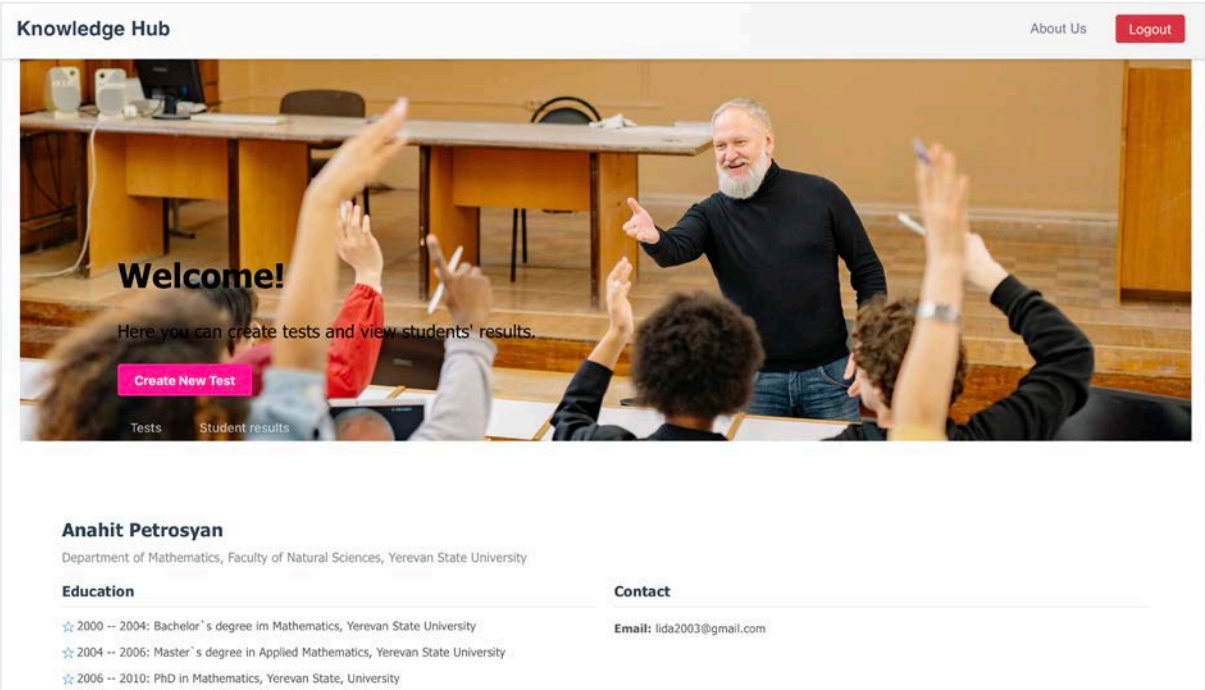


Նկ. 3.3. Վավերացման էջ

Դասախոսների մուտքը համակարգ նախատեսված է ավելի պարզեցված եղանակով. նրանց տրամադրվում է նախապես սահմանված գաղտնաբառ, որով նրանք անմիջապես կարող են մուտք գործել իրենց անձնական էջեր՝ առանց լրացուցիչ գրանցման անհրաժեշտության: Այս մոտեցումը հեշտացնում է դասախոսների աշխատանքը:

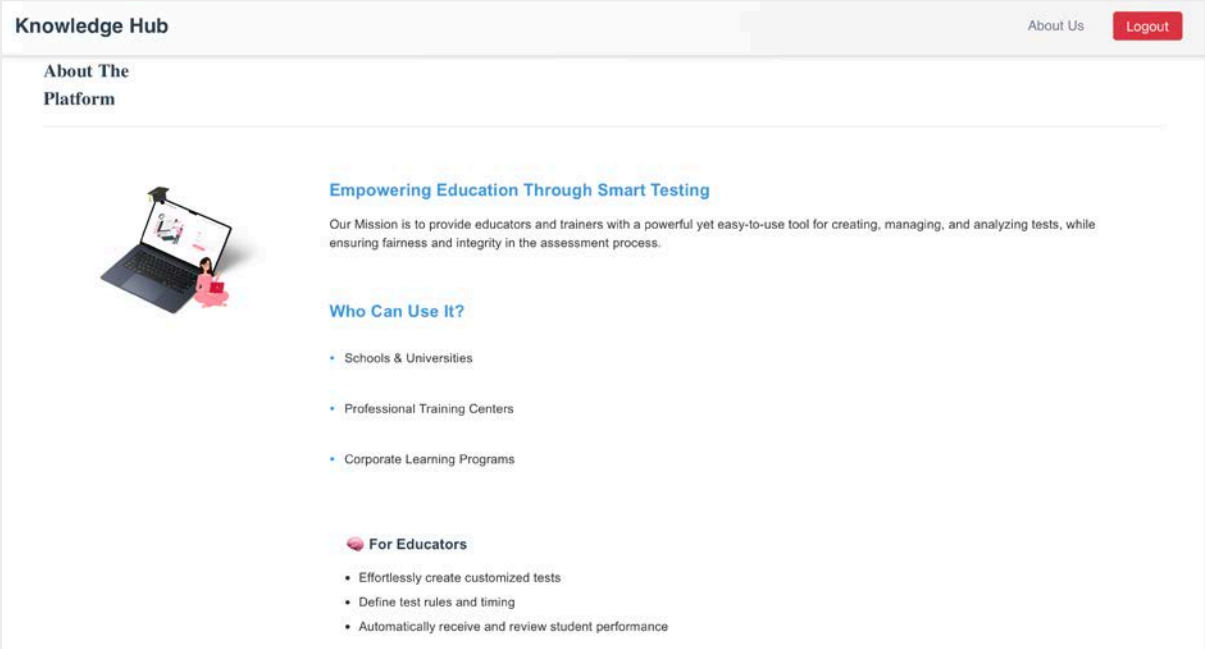
3.2 Համակարգի օգտատերերը եւ իրենց հնարավորությունները

Մշակված համակարգի օգտատերերը երկուսն են՝ Դասախոսներ եւ Ուսանողներ: Դասախոսների դերը կայանում է թեստային առաջադրանքների ստեղծման, կազմակերպման, մշակման, թարմացման մեջ: Դասախոսի մուտք գործելուց հետո բացվելու է հետեւյալ էջը



Նկ. 3.4. Դասախոսի էջ

Վերոնշյալ էջի Header-ը կազմված է About Us և LogOut բաժիններից, About Us բաժինը իրենից ներկայացնում է էջ, որտեղ նշվում է տեղեկություններ համակարգի մասին.



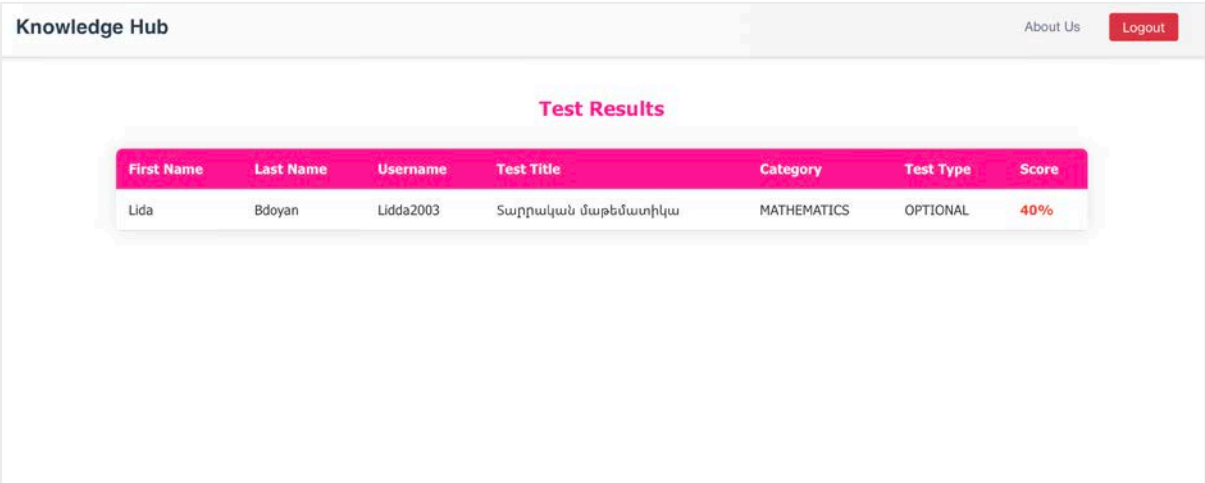
Նկ. 3.5 About Us էջ

Դասախոսի էջը պարունակում է նաև Create New Test, Tests և Student Results բաժինները:

Tests բաժնում դասախոսներին ներկայացվում է իրենց կողմից ստեղծված բոլոր թեստերի ցանկը: Այս էջում յուրաքանչյուր թեստի համար ցուցադրվում են հիմնական տվյալները, ինչպիսիք են թեստի անունը, տիպը, կատեգորիան: Դասախոսները կարող են արագորեն դիտել թեստերի ամբողջական ցանկը, ընտրել անհրաժեշտ թեստը՝ այն փոփոխելու կամ նոր թեստ ստեղծելու համար: Այս բաժինը նպաստում է թեստերի կառավարման գործընթացի կազմակերպվածությանը և արդյունավետությանը:

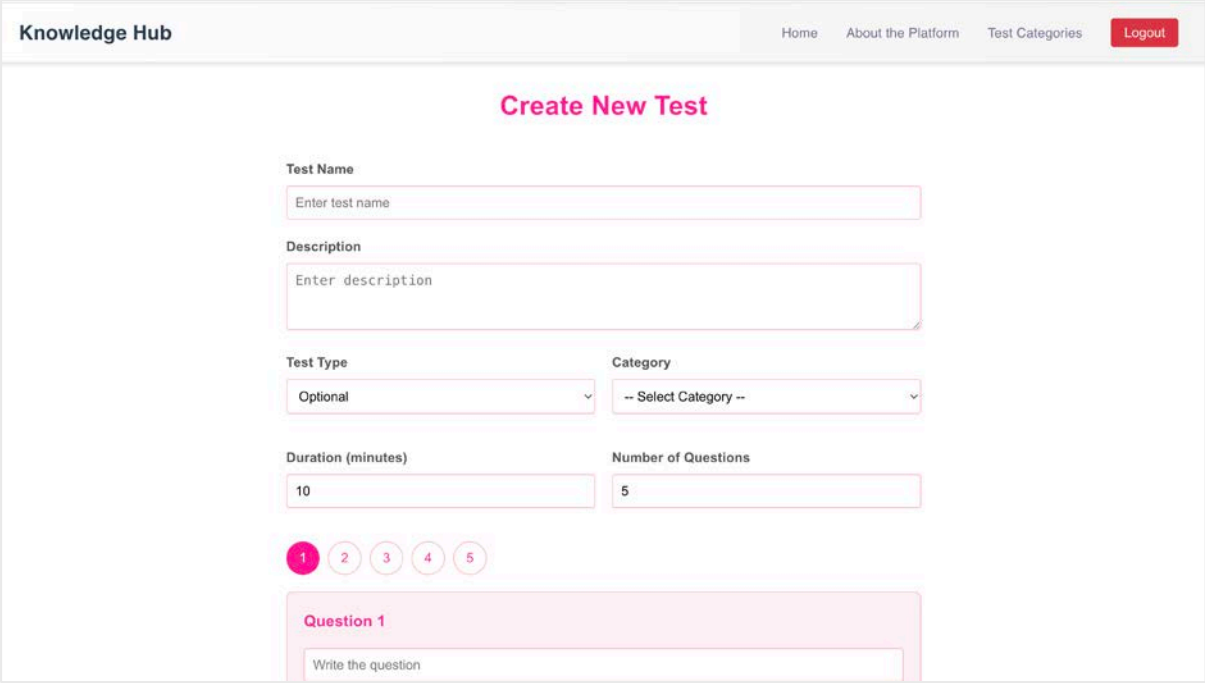
Նկ. 3.6. Դասախոսի կողմից ստեղծված թեստերի էջ

Student Results բաժնում դասախոսներին հասանելի են ուսանողների թեստավորման արդյունքները: Այս էջում ցուցադրվում են ուսանողների անունները, Username-ը, համապատասխան թեստի անվանումը, թեստի տեսակը, յուրաքանչյուր ուսանողի ստացած միավորը տոկոսներով:



Նկ. 3.7. Test Results էջ

Այժմ ուսումնասիրենք Create New Test բաժինը:



Նկ. 3.8 Create New Test էջ

Տվյալ էջում իրականացվում է թեստերի ստեղծման գործընթացը: Օգտատիրոջը (դասախոսին) հնարավորություն է տրվում սահմանել թեստի հիմնական պարամետրերը՝ ապահովելով թեստավորման ճշգրիտ և նպատակային կազմումը:

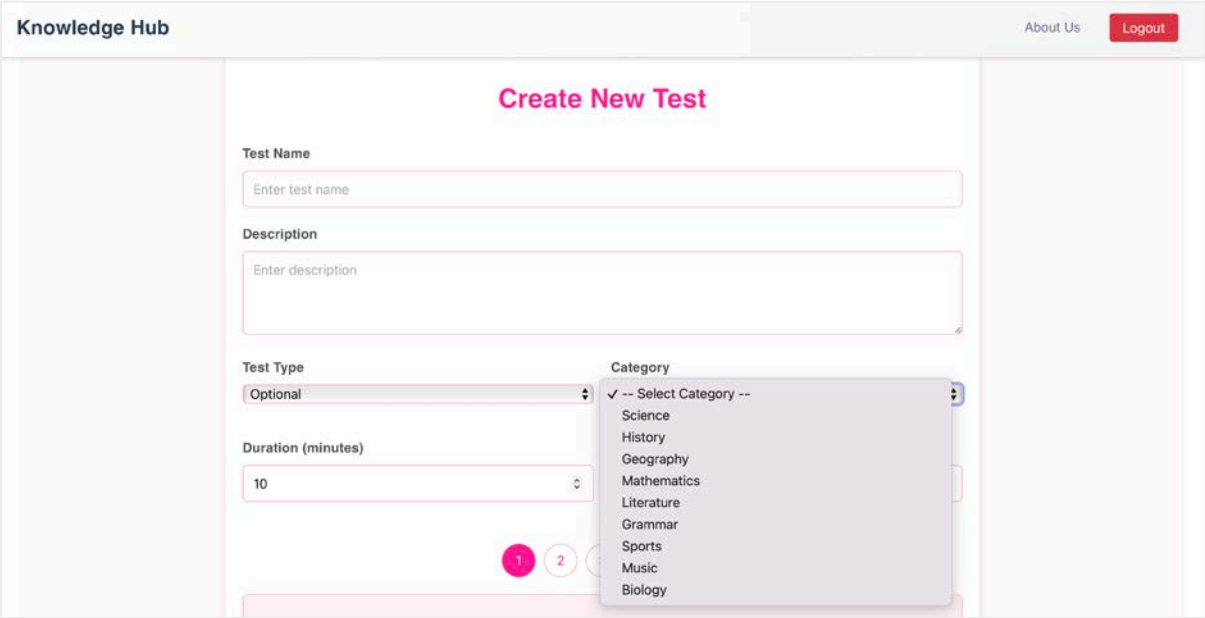
Առաջին հերթին, անհրաժեշտ է նշել **թեստի անվանումը** և տրամադրել **նկարագրություն**, որտեղ կբացատրվի, թե ինչ թեմայի կամ նյութի վերաբերյալ է տվյալ թեստը: Սա օգնում է ուսանողներին նախապես տեղեկանալ թեստի բովանդակության մասին:

Այնուհետև ընտրվում է **թեստի տիպը**, որը կարող է լինել՝

- **Optional** (ընտրովի պատասխաններով թեստ), երբ հարցերի հետ տրվում են տարբեր պատասխանների տարբերակներ, և ուսանողը պետք է ընտրի ճիշտ պատասխանը:
- **Written** (գրավոր պատասխաններով թեստ), երբ ուսանողը պետք է ինքնուրույն գրավոր ձևով պատասխան տա հարցերին:

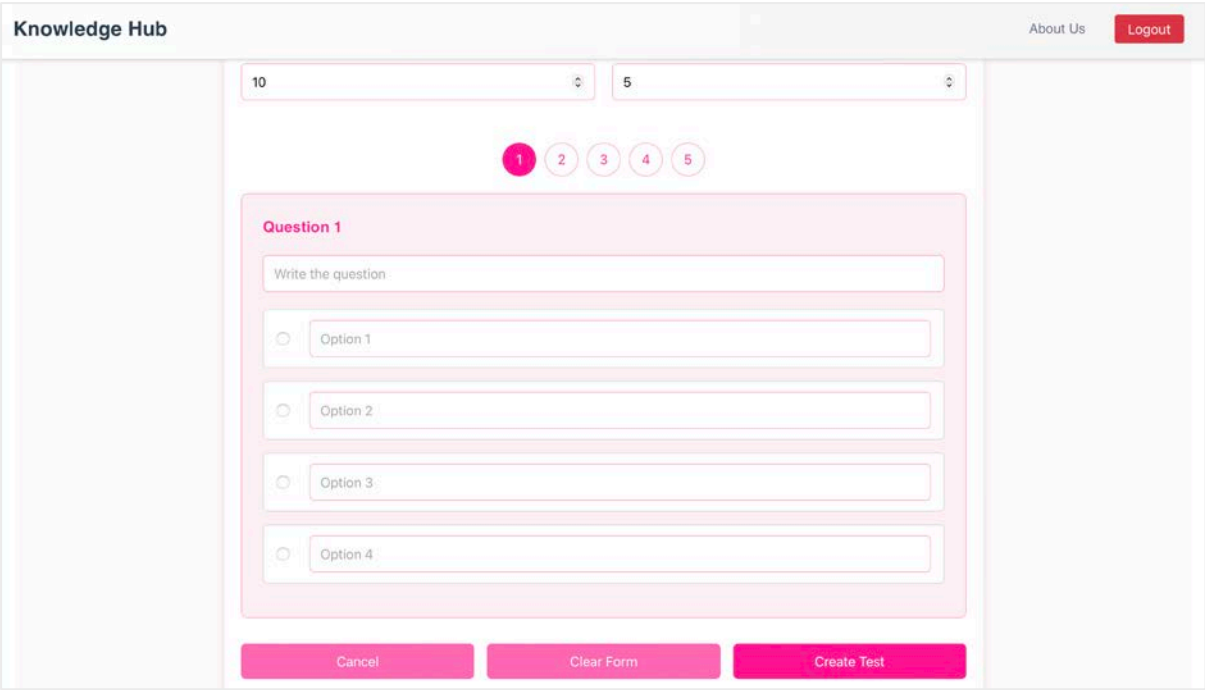
Հաջորդ քայլում ընտրվում է թեստի կատեգորիան, որը համապատասխանեցվում է դասավանդվող առարկային կամ թեմատիկ

ուղղությանը: Սա թույլ է տալիս դասակարգել թեստերը և հեշտացնել ուսանողների կողմնորոշումը:



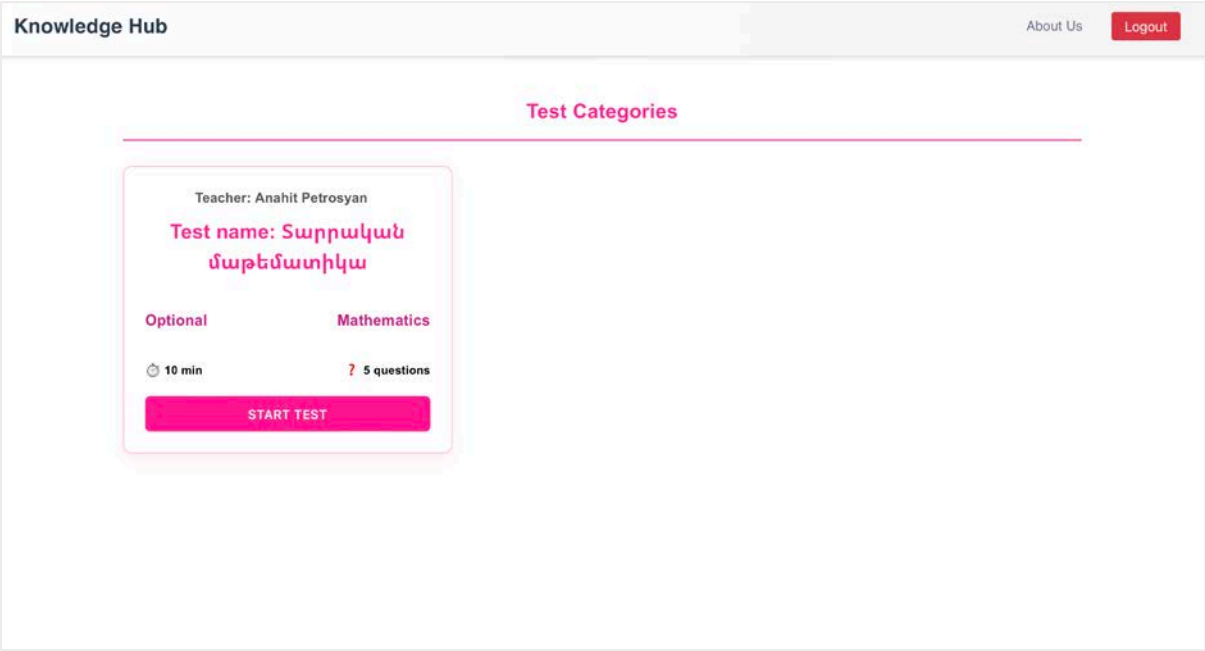
Նկ. 3.9 Create New Test էջ, թեստի կատեգորիայի ընտրում

Բացի այդ, սահմանվում է թեստի տևողությունը՝ րոպեներով, որը կարգավորում է, թե որքան ժամանակ կունենա ուսանողը թեստի բոլոր հարցերին պատասխանելու համար: Այնուհետև, նշվում է հարցերի ընդհանուր քանակը, ինչը սահմանում է թեստի ծավալը և բարդությունը:



Նկ. 3.10 Create New Test էջ, հարցերի ստեղծում

Այժմ ուսումնասիրենք ուսանողների մուտքը: Ուսանողների գրանցումից և հաջողությամբ մուտք գործելուց հետո բացվում է հետեւյալ էջը՝

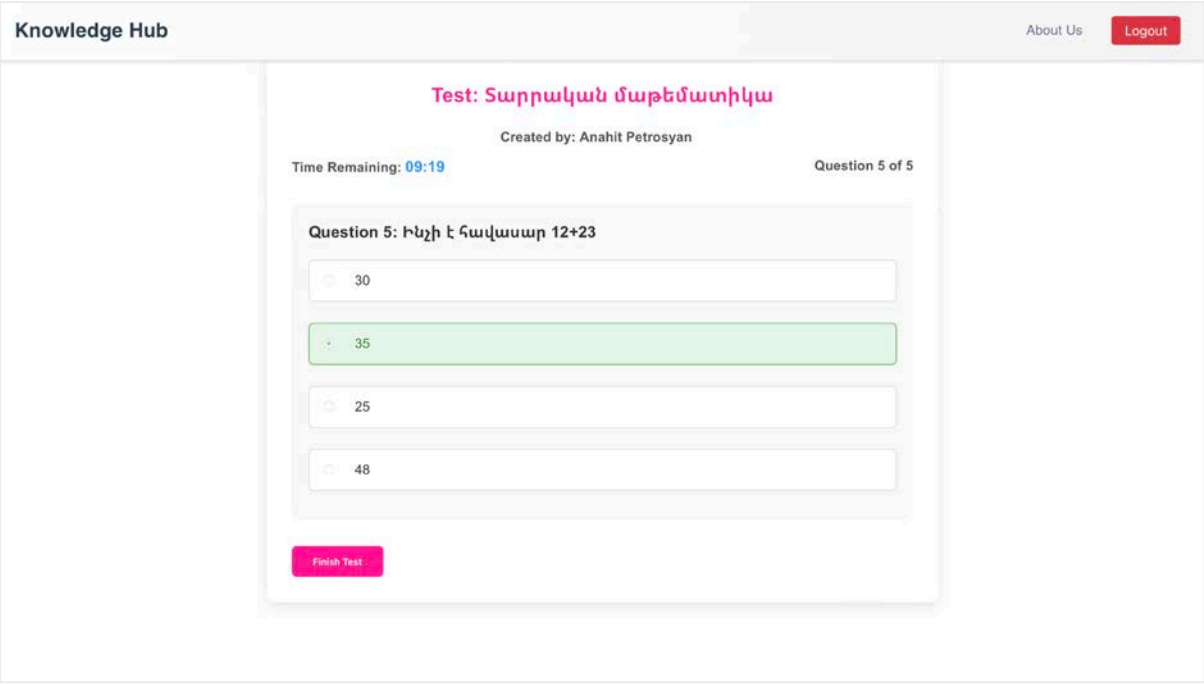


Նկ. 3.11 Ուսանողի էջ

Ուսանողի էջում ցուցադրվում է բոլոր հասանելի թեստերի ցանկը, որոնք ստեղծվել են տարբեր դասախոսների կողմից: Յուրաքանչյուր թեստի կողքին ներկայացված են թեստի վերաբերյալ հիմնական տեղեկատվական դաշտերը, մասնավորապես՝

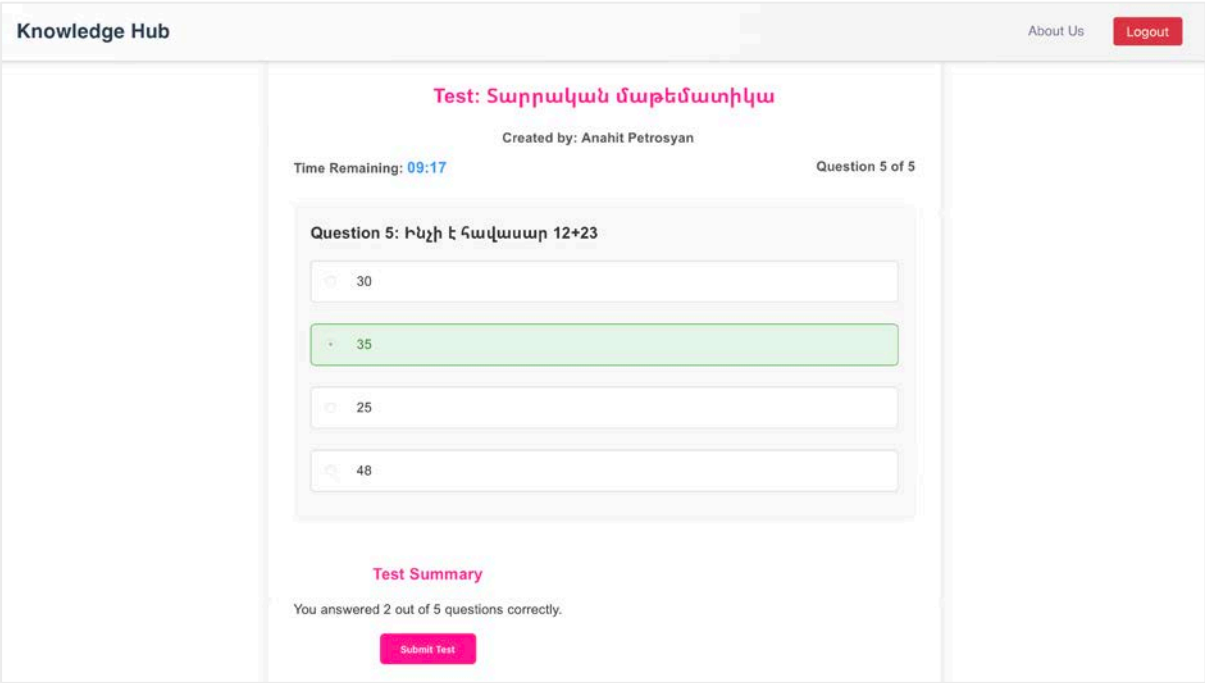
- Թեստի ստեղծող դասախոսի անունը,
- Թեստի անվանումը,
- Թեստի տիպը,
- Թեստի կատեգորիան,
- Թեստի տևողությունը՝ սահմանված րոպեների քանակով,
- Հարցերի ընդհանուր քանակը:

Այս տեղեկատվությունը օգնում է ուսանողին ընտրել համապատասխան թեստը՝ իր պատրաստվածության մակարդակի և հետաքրքրության ոլորտի համաձայն: Յուրաքանչյուր թեստի կողքին առկա է **"Start Test"** կոճակը, որի սեղմումից անմիջապես հետո սկսվում է թեստավորման գործընթացը: Թեստը սկսելուց հետո գործարկվում է նաև ժամանակաչափը՝ վերահսկելու համար տրված ժամանակը, և ուսանողը կարող է անցնել հարցերին պատասխանելուն:



Նկ. 3.12 Հարցերին պատասխանելու գործընթացը

Թեստի հարցերին պատասխանելու ընթացքում ուսանողը հերթականությամբ անցնում է հարցից հարց: Երբ ուսանողը հասնում է վերջին հարցին, էկրանի ներքի հատվածում հայտնվում է Finish Test կոճակը: Finish Test կոճակի սեղմումով ուսանողը ավարտում է թեստը, և համակարգը ավտոմատ կերպով հաշվարկում է նրա արդյունքները: Էկրանում ուսանողին ներկայացվում է տեղեկատվություն՝ թե որքան հարցից քանիսին է ճիշտ պատասխանել: Նաեւ ուսանողին հասանելի է դառնում **"Submit Test"** կոճակը, որի սեղմումով նա պաշտոնապես ամրագրում է իր թեստի պատասխանները համակարգում:

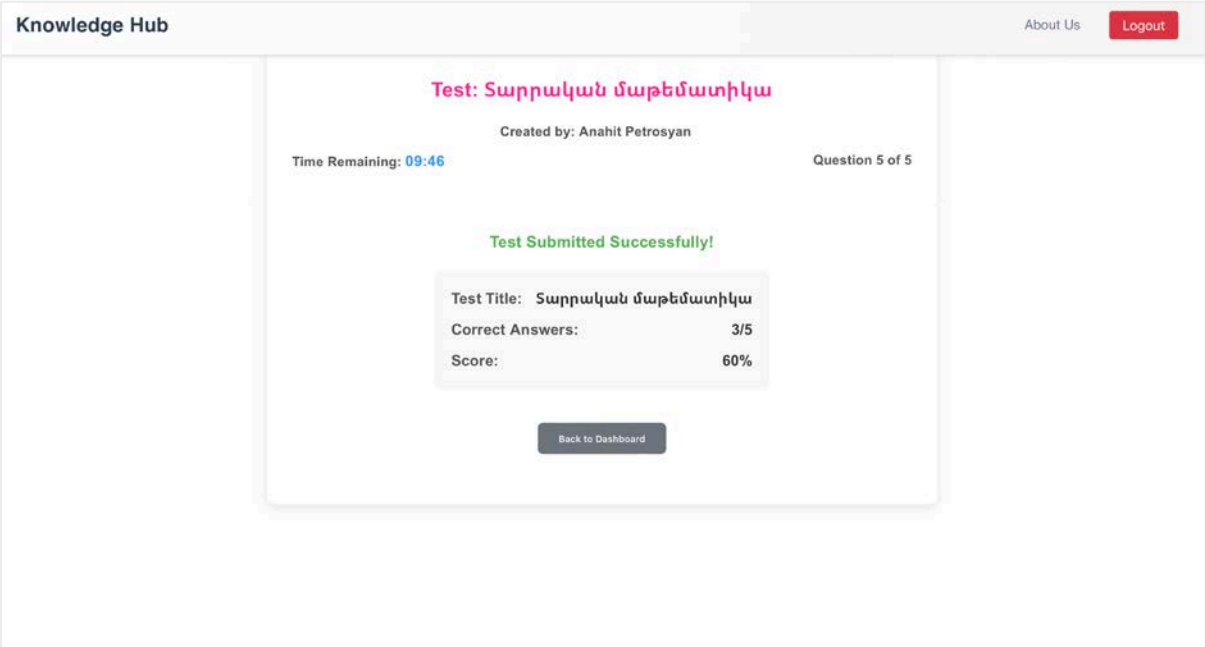


Նկ. 3.13 Թեստի ավարտ

Submit Test կոճակի սեղմումը հանդիսանում է թեստի հանձնման գործընթացի ավարտական փուլը: Սեղմելով այս կոճակը՝ ուսանողը պաշտոնապես հաստատում է իր պատասխանները, և թեստի արդյունքները պահպանվում են համակարգում: Տվյալները գրանցվում են ուսանողի անհատական **Student Results** բաժնում՝ հասանելի լինելով հետագա վերլուծության և դասախոսի կողմից գնահատման համար: Submit Test կոճակի սեղմումից հետո ավտոմատ կերպով

բացվում է արդյունքները, որտեղ ուսանողին ներկայացվում է իր թեստավորման մանրամասն հաշվետվությունը: Այս էջում տրամադրվում են հետևյալ տվյալները՝

- Թեստի ավարտման համար օգտագործված ժամանակահատվածը (թե որքան ժամանակում է ուսանողը այն ավարտել),
- Հարցերից քանիսին է ուսանողը տվել ճիշտ պատասխան,
- Արդյունքի տոկոսային արտահայտությունը, որը հաշվարկվում է ճիշտ պատասխանված հարցերի քանակի հիման վրա:



Նկ. 3.14 Թեստի արդյունքը

Այստեղ ուսանողին ներկայացվում է իր թեստավորման մանրամասն հաշվետվությունը: Այս էջում տրամադրվում են հետևյալ տվյալները՝

- Թեստի ավարտման համար օգտագործված ժամանակահատվածը (թե որքան ժամանակում է ուսանողը այն ավարտել),
- Հարցերից քանիսին է ուսանողը տվել ճիշտ պատասխան,
- Արդյունքի տոկոսային արտահայտությունը, որը հաշվարկվում է ճիշտ պատասխանված հարցերի քանակի հիման վրա:

Գլուխ 4. Տնտեսագիտություն

«Թեստավորման միջոցով գիտելիքների մակարդակի որոշման և ամփոփման գործիքամիջոցի մշակում» ծրագրային փաթեթի ինքնարժեքի և գնի հաշվարկը

Ժամանակակից տնտեսական պայմաններում ցանկացած գործունեություն առնչվում է տնտեսական հարցերի հետ: Նոր տեխնիկայի, տեխնոլոգիական պրոցեսների, տեխնիկա-տնտեսական հիմնավորման համար, մասնավորապես մեր աշխատանքի պայմաններում կարևորագույն տնտեսական հիմնահարցերից է ինքնարժեքի որոշումը:

Արտադրանքի կամ ծառայությունների ինքնարժեքը արտադրանքի (ծառայությունների) արտադրության և իրացման վրա կատարված բոլոր ծախսերի գումարն է դրամական արտահայտությամբ:

Ինքնարժեքի մեջ իրենց արտահայտությունն են գտնում սպառված շրջանառու ֆոնդերը, կենդանի աշխատանքի մի մասը, որը աշխատողներին վճարում է աշխատավարձի ձևով:

Ինքնարժեքի մեջ մտնող ծախսերը դասակարգվում են ըստ տնտեսական տարերի և ըստ կալկուլյացիոն հոդվածների:

Ներկայումս կիրառվում է ծախսերի ըստ կալկուլյացիոն հիմնական հոդվածների հետևյալ դասակարգումը՝

1. համալրող առարկաներ,
2. էլեկտրաէներգիայի ծախսեր,
3. աշխատողների հիմնական աշխատավարձ,
4. աշխատողների լրացուցիչ աշխատավարձ,
5. սարքավորումների շահագործման և պահպանման ծախսեր,
6. տարածքի վարձակալության համար ծախս,
7. այլ գործառնական ծախսեր:

ինքնարժեք (1-7 կետերի գումարը)։

Փաթեթը, որի ինքնարժեքն ու գինը ենթակա է որոշման, իրենից ներկայացնում է՝ XYZ ծրագրային փաթեթ։

Հաշվարկի համար ելքային տվյալներ են հանդիսանում՝

- փաթեթի մեջ մտնող համալրող առարկաների քանակն ու անվանացանկը;
- ժամանակի ամփոփ նորմերը, աշխատանքի կարգն ու աշխատավարձի ձևերը,
- ժամավճարային և գործարքային պարգևատրման չափերը (22 %),
- լրացուցիչ աշխատավարձի չափերը (12 %),
- սարքավորումների պահպանման և շահագործման ծախսերի դրույքաչափերը (2,2 %),
- ընդհանուր տնտեսական ծախսերի դրույքաչափը (4,4 %)։

1. Համալրող առարկաների ծախսի հաշվարկ

Գնված բաղադրիչների արժեքը որոշվում է հետևյալ բանաձևով՝

$$C_{пк} = \sum M_{пkj} * P_{пkj},$$

որտեղ՝ $M_{пkj}$ j-րդ տեսակի գնված բաղադրիչների քանակն է, հատ, $P_{пkj}$ j-րդ գնված բաղադրիչի գինը, դրամ։ Հաշվարկի արդյունքները բերված են աղյուսակ 1-ում։

Աղյուսակ *1.*

Բաղադրիչի անվանումն ու տեսակը	Մեկ փաթեթին ընկնող քանակ.հատ	Միավորի գինը.դրամ	Մեկ փաթեթին ընկնող արժեք.դրամ
IntelliJ IDEA Professional	1	30.000	30.000
Թղթապանակ	3	500	1.500

Ինտերնետ	1	5000	5000
Դոմեյն	1	50.000	50.000
Ընդամենը			86.500

Ընդամենը՝ 86.500

2. Էլեկտրաէներգիայի ծախսի հաշվարկ

Համակարգիչները և այլ սարքավորումները աշխատեցնելու համար էլեկտաէներգիայի ծախսը որոշվում է հետևյալ բանաձևով՝

$$Է=W*Uէ$$

որտեղ՝ W-ն էլեկտրաէներգիայի ամսեկան ծախսն է, Uէ-ն 1 կվ/ժ էլեկտրաէներգիայի արժեքն է (≈50 դրամ)։ Օրական ծախս՝ 0.015 կՎտ × 8 ժ = 0.12 կՎտ/ժ, ամսեկան ծախսը՝0.12 × 30 = 3.6 կՎտ/ժ:

Էլեկտրաէներգիայի ամսական ծախսը մոտավոր կկազմի W≈3.6կՎտ/ժ, Է=3.6*50=180 դրամ/ամսական և 180*5= 900 դրամ/3 ամսում:

(Գրասենյակում կա 1 համակարգիչ, յուրաքանչյուրը 15 Վտ W հզորությամբ, աշխատում են օրական 8 ժամ)

3. Աշխատողների հիմնական աշխատավարձի հաշվարկը

Աշխատողների հիմնական աշխատավարձի մեջ մտնում են՝

- գործարքային դրույքաչափերով աշխատավարձ,
- ժամավճարային աշխատավարձ,
- պարգևավճար:

Գործարքային աշխատավարձն ըստ տարիֆային համակարգի որոշվում է հետևյալ բանաձևով՝

$$Ահիմ.= Ժդ.*Ապրտ.,$$

որտեղ՝ $Ժ_դ$-ն ժամային դրույքաչափն է, $Ա_{արտ.}$-ն ՝ ժամային նորմը: Հաշվարկման արդյունքները բերված են աղյուսակ 2-ում:

Աղյուսակ *2*

Գործառույթի հաջորդական.	Վճարման ձև	Ժամանակային նորմ	Ժամային դրույք	Տարիֆային ֆոնդ
Նախապատրաստում	Գործարքա-պարգևավճարային	30	3500	105.000
Մշակում	Գործարքա-պարգևավճարային	30	5000	150.000
Կարգավորում	Գործարքա-պարգևավճարային	25	5000	125.000
Տեստավորում	Գործարքա-պարգևավճարային	25	4500	112.500
Ընդամենը				492.500

Պարգևատրման չափը որոշվում է հետևյալ բանաձևով՝

$$Պ= Ա_{հիմ} * Պ_դ/100\%,$$

որտեղ՝ $Պ_դ$ -պարգևատրման դրույքաչափ, %

Պ = 492.500 *22/100 % = 108.240 դրամ:

Ընդամենը հիմնական աշխատավարձը կկազմի՝

492.500+108.240=600.240 դրամ/ամսական կամ՝

600.240*5= 3,001,200 դրամ/5 ամսում:

1. **Աշխատողների լրացուցիչ աշխատավարձի հաշվարկը**

Լրացուցիչ աշխատավարձի մեջ մտնում են՝հերթական և լրացուցիչ գործողումների, արձակուրդների վճարները, պետական հանձնարարականների կատարման հետ կապված ծախսերը և այլն: Աշխատողների լրացուցիչ աշխատավարձը հաշվարկվում է հետևյալ բանաձևով՝

$$Ա_{\text{լր.}} = ԸԱ_{\text{հիմ.}} * Ա_{\text{լր.դ}}/100,$$

որտեղ՝ $ԸԱ_{\text{լր.դ}}$ – Ընդհանուր հիմնական աշխատավարձն է, իսկ $Ա_{\text{լր.դ}}$-ն լրացուցիչ աշխատավարձի դրույքաչափն է,%:

$$Ա_{\text{լր.}} = 600.240*12/100 = 72.028 \text{ դրամ/ամսական կամ}$$

$$72.028*5 = 360,144 \text{ դրամ/5 ամսում:}$$

5. Սարքավորումների պահպանման և շահագործման ծախսերի հաշվարկը

Սարքավորումների պահպանման և շահագործման ծախսերի թվին են պատկանում ամորտիզացիոն, ընթացիկ վերանորոգման, տրանզիտորային միջոցների, գործիքների և հարմարանքների վերանորոգման և այլ ծախսերը: Սարքավորումների պահպանման և շահագործման ծախսերի թվին են պատկանում ամորտիզացիոն, ընթացիկ վերանորոգման, տրանզիստորային միջոցների, գործիքների և հարմարանքների վերանորոգման և այլ ծախսերը:

5.1 Հիմնական միջոցների ամորտիզացիա

Հիմնական միջոցների տարեկան ամորտիզացիան ($Ա_{տ}$) հաշվարկվում է հետևյալ բանաձևով՝

$$Ա_{տ}=Հ_{ա}/Ն,$$

որտեղ $Հ_{ա}$-ն հիմնական միջոցների սկզբնական արժեքն է, Ն-ն հիմնական միջոցների օգտակար գործունեության ժամկետն է:

Աղյուսակ *3.*

		Ամորտիզացիոն հատկացումներ

Հիմնական միջոցի անվանումն ու տեսակը	Հիմնական միջոցի սկզբնական արժեքը	ՀՄ օգտակար գործողության ժամկետ, տարի	Ամորտիզացիոն ծախս
Սարքավորումներ, շենքեր, այլ հիմն. միջոցներ	3.165.000	5	633.000
Ընդամենը			

Սարքավորումների պահպանման և շահագործման ծախսերի գումարը որոշվում է հետևյալ կերպ՝

$$Ծ_{պ.շ.} = Ա_{հիմ}*5 * Ծ_{պ.շ.դ.}/100,$$

որտեղ՝ $Ծ_{պ.շ.դ.}$ - սարքավորումների պահպանման և շահագործման ծախսերի դրույքաչափն է, $Ա_{հիմ.}$–աշխատողների տարեկան հիմնական աշխատավարձը:

$$Ծ_{պ.շ.} = 492.500*5*2.2/100 = 54,175 \text{ դրամ:}$$

Սարքավորումների ընթացիկ վերանորոգման ծախսը որոշվում է հետևյալ բանաձևով՝

$$Ծ_{ը.վ.} = Ա_{հիմ} *5* Ծ_{ը.վ.դ.}/100,$$

որտեղ՝ $Ծ_{ը.վ.դ.}$ - սարքավորումների ընթացիք վերանորոգման դրույքաչափն է:

$$Ծ_{ը.վ.} = 492.500*5*4.4/100 = 108,350 \text{ դրամ:}$$

Աղյուսակ *4.*

Ծախսի անվանումը	Դրույքաչափը	Տարեկան ծախսը
Սարքավորումների պահպանման և շահագործման ծախսեր	2.2	54,175
Սարքավորումների ընթացիկ վերանորոգում	4.4	108,350
Սարքավորումների ամորտիզացիա	-	633.000

Ընդամենը	795,525

6. Տարածքի վարձակալության ծախսի հաշվարկը

Գործունեության արդյունավետության հիմնական պայմաններից մեկը տարածքի ճիշտ ընտրությունն է: Վարձակալվող տարածքը կգտնվի քաղաք Երևանում (Քանաքեռ Զեյթուն): Այս տարածքում անհրաժեշտ տարածքի վարձակալության ամսեկան ծախսը կկազմի 50.000 դրամ կամ 50.000*5 = 250.000 դրամ տարեկան:

7. Գործառնական այլ ծախսերի հաշվարկը

Գործառնական այլ ծախսերի մեջ մտում են ձեռնարկության ընդհանուր կառավարման-ադմինիստրատիվ՝ ադմինիստրատիվ անձնակազմի աշխատավարձի, գործուղման, գովազդային և այլ ծախսեր: Այն որոշվում է հետևյալ բանաձևով՝

$$ԸՏԾ = Ա_{հիմ} * \%ԸՏԾ/100,$$

որտեղ՝ %ԸՏԾ - Ընդհանուր տնտեսավարման ծախսերի տոկոսն է %:

ԸՏԾ = 3,001,200 *117/100 = 3,511,404 դրամ:

8. Փաթեթի ընդհանուր ինքնարժքի կալկուլացիան

Ինքնարժեքի կալկուլացիան բերված է աղյուսակ 5-ում:

Աղյուսակ *5.*

N	Ծախսերի հոդվածի անվանումը	գումարը, դրամ
1.	Համալրող առարկաներ	86.500
2.	Էլեկտրաէներգիա	900
3.	Ընդհանուր հիմնական աշխատավարձ	3,001,200

4.	Լրացուցիչ աշխատավարձ	360,144
5.	Սարքավորումների պահպանման և շահագործման ծախսեր	54,175
6.	Տարածքի վարձակալություն	50.000
7.	Գործառնական այլ	3,511,404
	ինքնարժեք	7,064,323

«Թեստավորման միջոցով գիտելիքների մակարդակի որոշման և ամփոփման գործիքամիջոցի մշակում» փաթեթի ինքնարժեքը կկազմի՝ Ի= 7,064,323 դրամ:

9. Շահույթի և միավորի գնի հաշվարկը

Փաթեթի գինն իր մեջ ընդգրկում է ընկերության շահույթն ու ինքնարժեքը: Շահույթի հաշվարկն իրականացվում է հետևյալ բանաձևով՝

Շ = Ի *%Շ/100,

որտեղ՝ Ի– փաթեթի ինքնարժեքն է, %Շ –շահույթի դրույքաչափը, %

Շ = 7,064,323 *18/100 = 1,271,578.14 դրամ:

Գնի հաշվարկը իրականացվում է հետևյալ կերպ՝

Գ = Ի+ Շ,

Գ = 7,064,323 + 1,271,578.14 = 8,335,901.14 դրամ:

Փաթեթի բացթողնման գինը որոշվում է հետևյալ բանաձևով՝

$Գ_{բաց.}$ = Գ + ԱԱՀ,

որտեղ՝ $Գ_{բաց.}$– փաթեթի բացթողնման գինն է, ԱԱՀ-ն ավելացված արժեքի հարկը (20%):

$Գ_{բաց.}$ = 8,335,901.14 + 8,335,901.14 *20/100 = 10,003,081.37 դրամ:

Այսպիսով, կատարված հաշվարկների արդյունքում ստացանք, որ «Թեստավորման միջոցով գիտելիքների մակարդակի որոշման և ամփոփման

գործիքամիջոցի մշակում» փաթեթի ինքնարժեքը կկազմի 8,359,480.14 դրամ, իսկ գինը՝ 10,031,376.17 դրամ:

Գլուխ 5. Կենսագործունեության անվտանգություն Համակարգչային սրահում օդափոխության կազմակերպում

Տեղեկատվական տեխնոլոգիաների զարգացումը մարդկային գործունեության գրեթե բոլոր ոլորտներում բերել է այն իրավիճակին, երբ համակարգչային տեխնիկան դարձել է անբաժանելի մաս ինչպես կրթական, այնպես էլ աշխատանքային գործընթացներում: Հատկապես կրթական հաստատություններում՝ դպրոցներում, համալսարաններում և ուսումնական կենտրոններում, համակարգչային սրահները կենտրոնական դեր են կատարում՝ նպաստելով ուսանողների և աշակերտների տեխնիկական գիտելիքների զարգացմանը:

Այս սրահներում սովորաբար տեղակայված են բազմաթիվ համակարգիչներ, տպիչներ, սքաներներ և այլ սարքավորումներ, որոնք երկարատև աշխատանքից հետո զգալիորեն բարձրացնում են սրահի ջերմաստիճանը և առաջացնում են օդի աղտոտում՝ ինչպես քիմիական, այնպես էլ մանրէաբանական մակարդակով: Բացի այդ, նույն տարածքում երկար ժամանակ գտնվում են մարդիկ, ինչը հանգեցնում է ածխաթթու գազի (CO_2) կուտակման, խոնավության ավելացման և օդի թթվածնային մակարդակի նվազման:

Բոլոր այս գործոնները կարող են բացասաբար ազդել ոչ միայն սարքավորումների աշխատանքի արդյունավետության վրա, այլև մարդկանց առողջության վրա: Այդ պատճառով համակարգչային սրահներում օդափոխության ճիշտ և արդյունավետ կազմակերպումը դառնում է կենսական անհրաժեշտություն: Օդափոխության համակարգերը ոչ միայն նպաստում են օդի թարմացմանը և մաքրությանը, այլև ապահովում են ջերմաստիճանի և խոնավության վերահսկումը, ինչը չափազանց կարևոր է կենսաբանական անվտանգության պահպանման համար:

Կենսաբանական անվտանգությունը ենթադրում է այնպիսի միջավայրի ապահովում, որտեղ նվազեցված են վարակների տարածման, ալերգիկ ռեակցիաների և առողջական այլ խնդիրների առաջացման ռիսկերն ու գործոնները:

Միշտ պետք է հաշվի առնել, որ համակարգչային սրահներում օդափոխության անբավարարությունն ու սխալ կազմակերպվածությունը կարող են հանգեցնել սթրեսի, հոգնածության, ուշադրության նվազման, ինչպես նաև վարակի տարածման:

Համակարգչային սրահները հիմնականում փակ տարածքներ են, որտեղ տեղակայված են բազմաթիվ համակարգիչներ, մոնիտորներ, տպիչներ և այլ տեխնիկական սարքեր: Սովորաբար սրահում առկա են նաև ուսանողներ, աշխատակիցներ կամ օգտվողներ, ովքեր երկար ժամանակ գտնվում են այդ տարածքում:Այս բոլոր գործոնները առաջացնում են. Ջերմության ավելացում, ածխաթթու գազի (CO_2) բարձր մակարդակ, հեղուկների եւ օդափոխիչների աշխատանքից առաջացած խոնավություն, մեղմ բայց կայուն աղտոտվածություն՝ փոշի, մանրէներ, քիմիական նյութեր:

Օդափոխությունը մեծ կարեւորություն ունի կենսաբանական անվտանգության տեսանկյունից: Կենսաբանական անվտանգությունը ենթադրում է այնպիսի պայմանների ստեղծում, որոնք նվազեցնում են առողջությանը սպառնացող ռիսկերն ու ապահովում են հիգիենիկ միջավայր: Օդափոխության բացակայությունը կամ ոչ ճիշտ կազմակերպումը կարող է բերել հետևյալ խնդիրներին՝ օդի որակի վատթարացում, կենտրոնացված CO_2-ի բարձրացում, որը հանգեցնում է քնի, հոգնածության և ուշադրության նվազման, բակտերիաների եւ վիրուսների արագ տարածում, ալերգիկ ռեակցիաներ եւ շնչառական խնդիրներ:

Ուստի օդափոխությունը նպաստում է մարդկանց առողջության պահպանմանը ինչպես նաև համակարգիչների աշխատանքային պայմանների օպտիմալացմանը և շրջակա միջավայրի բարելավմանը:

Համակարգչային սրահներում կիրառվում են հիմնականում երկու տեսակի օդափոխություն՝ **բնական օդափոխություն,** սա օդի շարժման բնական գործընթացն է՝ դռների, պատուհանների միջոցով: Այն կարող է արդյունավետ լինել միայն այն դեպքում, երբ սրահը ունի պատուհաններ և բավարար հոսքեր: Բնական օդափոխության առավելությունն այն է, որ չի պահանջում տեխնիկական սարքավորումներ և էժան , հասանելի միջոց է: Իսկ թերությունը կայանում է

նրանում, որ կախված է արտաքին պայմաններից և անհնար է վերահսկել օդի շարժման ինտենսիվությունը։

Օդափոխության հաջորդ տեսակը մեխանիկական օդափոխությունն է, որը ապահովվում է օդափոխման համակարգերի միջոցով՝ օդափոխիչներ, օդափոխման խողովակներ, օդորակիչներ։

Մեխանիկական օդափոխության առավելությունը կայուն օդափոխությունն է՝ անկախ եղանակից և հնարավորություն է կարգավորելու օդի ջերմաստիճանը, խոնավությունը, մաքրությունը։ Իսկ թերությունը տեղադրման և պահպանման բարձր արժեքը և էլեկտրաէներգիայի սպառման մեծացումն է։

Օդափոխության համակարգի նախագծումը համակարգչային սրահի համար մի գործընթաց է, որը պահանջում է մանրակրկիտ պլանավորում և համապատասխանեցում մի շարք բնութագրիչների հետ։ Նպատակը՝ ապահովել այնպիսի պայմաններ, որոնք կնպաստեն մարդկանց առողջ աշխատանքին և տեխնիկական սարքավորումների անխափան օգտագործմանը։ Եթե օդափոխությունը սխալ է նախագծված կամ ընդհանրապես բացակայում է, ապա դա կարող է հանգեցնել ոչ միայն առողջական խնդիրների, այլ նաև սարքավորումների խափանման, ինչը հատկապես զգայուն է կրթական կամ արտադրական հաստատությունների համար։

Օդափոխության նախագծումը պետք է սկսվի սրահի մանրակրկիտ վերլուծությամբ։ Դրան պետք է նախորդի սրահի մակերեսի և ծավալի (մ³) գնահատումը։ Օդափոխության համակարգը պետք է համապատասխանի սրահի տարածքին՝ ապահովելով բավարար օդափոխություն։ Ինչպես նաև համակարգիչների և այլ սարքավորումների քանակի գնահատումը, սարքավորումները արտազատում են ջերմություն, փոշի, երբեմն՝ օզոն, ինչը պահանջում է ուժեղացված օդափոխություն։ Մարդիկ նույնպես ջերմություն են արտազատում, ինչպես նաև՝ ածխաթթու գազ այդ իսկ պատճառով անհրաժեշտ է նաև օգտագործողների միջին քանակի գնահատում։ Սրահում մարդկանց քանակի աճը մեծացնում է թթվածնի սպառումը։ Եթե սրահն աշխատում է մի քանի ժամ

օրական կամ նույնիսկ ամբողջ օրը, օդափոխությունը պետք է լինի անընդհատ ու կանոնավոր և անհրաժեշտ է գնահատել օգտագործման տևողությունը։

Համակարգչային սրահի համար առավել նպատակահարմար է մեխանիկական օդափոխությունը, որը թույլ է տալիս վերահսկել օդի հոսքի ծավալը, ջերմաստիճանը և մաքրությունը։ Դրա մեջ կարող են ընդգրկվել հետևյալ տարրերը՝ վենտիլյացիոն համակարգեր՝ մուտքային և ելքային հոսքերի համար, օդորակիչներ՝ ջերմաստիճանի կարգավորման համար, արդիական օդի ֆիլտրեր՝ HEPA կամ ածխային ֆիլտրեր՝ փոշին, մանրէներն ու ալերգենները զտելու համար, օդի խոնավացուցիչներ կամ չորացուցիչներ՝ խոնավության մակարդակը հավասարակշռելու համար։

Նախագծման ժամանակ պետք է հաշվի առնել նաև հոսքագծերի ճիշտ տեղաբաշխումը։ Օդը պետք է հավասարաչափ շրջանառվի սրահում՝ խուսափելու համար "մեռած գոտիներից", որտեղ օդը չի շարժվում։

Կան որոշ ստանդարտներ և ցուցանիշներ, որոնց պետք է համապատասխանել նախագծման ընթացքում. Սրահում օդի ամբողջ ծավալը պետք է փոխարինվի առնվազն 4-6 անգամ յուրաքանչյուր ժամում, թթվածնի պարունակությունը պետք է պահել ≥ 19.5%։ Պակասը կարող է առաջացնել գլխացավ, թուլություն, **ածխաթթու** գազի մակարդակ պետք է լինի ≤ 1000 ppm։ Ավելը համարվում է առողջության համար վնասակար։ Ջերմաստիճանը՝ 21°C - 24°C՝ կախված տարվա եղանակից։ Խոնավությունը՝ 40%-60% միջակայքում։

Քանի որ համակարգչային սրահներում սովորաբար պահանջվում է լռություն՝ դասընթացների կամ կենտրոնացված աշխատանքի համար, անհրաժեշտ է օգտագործել աղմուկը նվազեցնող օդափոխիչներ կամ դրանք տեղադրել պատերի ներսում՝ կիրառելով ձայնամեկուսիչ նյութեր։

Օդափոխության համակարգը պետք է պարբերաբար ստուգվի և սպասարկվի, որպեսզի այն ապահովի նույն արդյունավետությունն ու մաքրությունը։ Ֆիլտրերը պետք է մաքրվեն կամ փոխարինվեն՝ ըստ արտադրողի հրահանգների։ Հակառակ դեպքում դրանք կարող են դառնալ փոշու ու մանրէների կուտակման աղբյուր։

Օդը պետք է մուտք գործի վերևից (կամ կողքից) և դուրս գա ներքևից կամ հակառակ կողմից։ Սա ապահովում է օդի հավասարաչափ շրջանառություն և նպաստում է ավելի արդյունավետ զտման գործընթացին։

Այսպիսով համակարգչային սրահներում օդափոխության համակարգի արդյունավետ կազմակերպումը հանդիսանում է ոչ միայն տեխնիկական պայմանների ապահովման միջոց, այլ նաեւ կենսաբանական անվտանգության առանցքային բաղադրիչ։ Այն նպաստում է առողջ, հիգիենիկ և հարմարավետ աշխատանքային միջավայրի ձևավորմանը՝ միաժամանակ կանխելով օդով փոխանցվող վարակների տարածումը, նվազեցնելով ալերգիկ և շնչառական խնդիրների առաջացման հավանականությունը, ինչպես նաև խթանելով կենտրոնացվածության և արդյունավետության բարձրացումը։

Օդափոխության համակարգի առկայությունը և նրա կանոնավոր սպասարկումը էական նշանակություն ունեն ինչպես մարդկանց առողջության պահպանման, այնպես էլ տեխնիկական սարքավորումների շահագործման ժամկետի երկարացման տեսանկյունից։ Բազմաթիվ հետազոտություններ ցույց են տվել, որ օդի որակը ուղղակիորեն ազդում է մարդու ճանաչողական գործառույթների վրա։ Բացի ֆիզիոլոգիական ազդեցությունից, օդի մաքրությունը կարող է նպաստել նաև բարոյահոգեբանական վիճակի բարելավմանը։ Մաքուր և թարմ օդով հագեցած սրահում աշխատողները հաճախ ունեն ավելի բարձր մոտիվացիա, կենտրոնացվածություն և գոհունակություն աշխատանքային պայմաններից։ Օդում ածխաթթու գազի բարձր մակարդակը կարող է նվազեցնել հիշողությունը, որոշումներ կայացնելու ունակությունը և ուշադրության կենտրոնացումը։ Հետևաբար, համակարգչային սրահների նախագծման և շահագործման փուլերում անհրաժեշտ է առաջնահերթություն տալ օդափոխության համակարգի ճիշտ ընտրությանը, մասնագիտական նախագծմանը և տեխնիկական սպասարկմանը՝ ապահովելով օդի որակի բարձր մակարդակ և կայուն աշխատանքային պայմաններ։

Գլուխ 6. Բնապահպանություն

Համակարգչային տեխնիկայի ազդեցությունը շրջակա միջավայրի վրա

Ժամանակակից աշխարհում տեղեկատվական տեխնոլոգիաների արագ զարգացումը նպաստում է էլեկտրոնային սարքերի, մասնավորապես՝ համակարգիչների լայն կիրառմանը: Սակայն դրանց արտադրությունը, շահագործումն ու անորակ թափոնների կառավարումը էական ազդեցություն ունեն շրջակա միջավայրի վրա: Համակարգչային տեխնիկայի արտադրությունը պահանջում է մեծ քանակությամբ բնական ռեսուրսներ, էներգիա և քիմիական նյութեր, որոնք օգտագործվում են տարբեր բաղադրիչների արտադրության ընթացքում:

Ներկայացնենք համակարգչային տեխնիկայի էկոլոգիական անվտանգությունը, նրա բացասական ազդեցության նվազեցման ուղիները և հնարավոր լուծումները: Մասնավորապես, նշենք թափոնների կառավարման մեթոդները, վերամշակման կարևորությունը և նորարարական տեխնոլոգիաների դերը բնապահպանական խնդիրների լուծման գործում:

Համակարգչային տեխնիկայի օգտագործումը դարձել է անբաժան մաս թե՛ կենցաղային, թե՛ արդյունաբերական ոլորտում: Սակայն դրա արտադրությունն ու շահագործումը ուղեկցվում են բնության վրա բացասական ազդեցությամբ, ինչը դառնում է արդիական բնապահպանական խնդիր:

Համակարգչային տեխնիկան իր ամբողջ կյանքի ցիկլի ընթացքում՝ սկսած հումքի արդյունահանումից մինչև շահագործում ու հետո վերամշակում, ունի տարբեր ձևերի բացասական ազդեցություն.

- **Օդային աղտոտում՝** արտադրության ժամանակ վնասակար գազերի արտանետում:
- **Ջրային աղտոտում՝** էլեկտրոնային բաղադրիչների լվացման գործընթացում մուտքային թունավոր նյութերի ներթափանցում:

- **Հողային ռեսուրսի աղտոտում՝** ոչ պատշաճ վերամշակման արդյունքում թունավոր մետաղների արտահոսք:
- **Էլեկտրաէներգիայի սպառում՝** շահագործման ընթացքում բարձր էներգիայի սպառում (հատկապես տվյալների կենտրոններում):

Համակարգիչների արտադրության և շահագործման հիմնական էկոլոգիական ռիսկերը կապված են հետևյալ գործոնների հետ.

- **Ռեսուրսների սպառում**

Համակարգիչների արտադրության համար անհրաժեշտ են հազվագյուտ մետաղներ, ինչպիսիք են պալադիումը, ոսկին, պլատինը և լիթիումը: Այս նյութերի արդյունահանումը մեծ վնաս է հասցնում էկոհամակարգերին, քանի որ հանքարդյունաբերությունը հանգեցնում է հողի էրոզիայի, ջրի աղտոտման և կենսաբազմազանության կորստի:

- **Էներգիայի սպառում**

Համակարգիչները և տվյալների կենտրոնները (data centers) էներգիայի մեծ սպառողներ են: Համաձայն ուսումնասիրությունների, միայն տվյալների կենտրոններն ամբողջ աշխարհում սպառում են շուրջ 200 տերավատ/ժամ (TWh) էներգիա տարեկան, ինչը համեմատելի է միջին զարգացած երկրի էներգետիկ սպառման հետ: Արդյունքում մեծանում է ածխաթթու գազի արտանետումների մակարդակը, ինչը նպաստում է կլիմայական փոփոխություններին:

- **Թափոնների կուտակում**

Տեխնոլոգիաների արագ զարգացումը հանգեցնում է նրան, որ սարքերը շատ արագ հնանում են և վերածվում թափոնների: Համաձայն ՄԱԿ-ի տվյալների, 2021 թվականին աշխարհում արտադրվել է շուրջ 57 միլիոն տոննա էլեկտրոնային թափոն (e-waste), որից միայն 20%-ն է վերամշակվել:

➢ **Բնապահպանական ազդեցության գնահատման մեթոդներ**

Համակարգչային տեխնիկայի բնապահպանական գնահատումը իրականացվում է հետևյալ հիմնական փուլերով.

1. Կատեգորիայի որոշում

Նախ անհրաժեշտ է որոշել, թե տեխնիկան ինչպես է ազդում շրջակա միջավայրի վրա.

- Ֆիզիկական ազդեցություն (աղմուկ, ճառագայթում),
- Քիմիական ազդեցություն (թունավոր նյութեր),
- Կենսաբանական ազդեցություն (ազդեցություն բուսական և կենդանական աշխարհին):

1. Որակական և քանակական վերլուծություն

Օգտագործվում են հետևյալ գործիքները՝

- Սահմանափակիչ ցուցանիշներ (ՄՀԿ-Մաքսիմալ հասանելի կոնցենտրացիաներ)
- Համակարգային գնահատում՝ ազդեցության աղբյուրի, տարածության և տևողության հաշվառումով
- Ինտեգրալ գնահատում՝ տարբեր գործոնների կշռված համադրման միջոցով

1. Վնասի գնահատում ըստ տարրական բաղադրիչների՝ որոշվում են վնասական տարրերը

Օրինակ՝ համակարգչային տեխնիկայի միկրոսխեմաներում առկա կապարն ու կադմիումը երկարաժամկետ ազդեցություն ունեն հողի և ստորգետնյա ջրերի վրա: Ըստ հեղինակների, նման ազդեցությունները պետք է ներառել որպես առաջնահերթ:

- **Էլեկտրոնային թափոնների խնդիրները**

Էլեկտրոնային թափոնները (e-waste) ներկայումս համարվում են աշխարհի ամենաարագ աճող թափոնային հոսքերից մեկը: Դրանց կառավարումը լուրջ խնդիր է ստեղծում ինչպես զարգացած, այնպես էլ զարգացող երկրներում:

Էլեկտրոնային թափոնները պարունակում են բազմաթիվ թունավոր նյութեր՝ մասնավորապես ծանր մետաղներ, որոնք վնասակար են ինչպես շրջակա միջավայրի, այնպես էլ մարդու առողջության համար: Օրինակ՝

- **Կապար (Pb)**՝ վնասում է նյարդային համակարգը, կարող է առաջացնել արյան հիվանդություններ,

- **Սնդիկ (Hg)**՝ ազդում է ներքին օրգանների վրա, մասնավորապես՝ լյարդի և երիկամների վրա,
- **Կադմիում (Cd)**՝ կարող է կուտակվել օրգանիզմում և առաջացնել քաղցկեղային հիվանդություններ,
- **Պլաստիկ բաղադրիչներ**՝ պարունակում են քիմիական նյութեր, որոնք երկար ժամանակ չեն քայքայվում և աղտոտում են շրջակա միջավայրը:

Բազմաթիվ երկրներում էլեկտրոնային թափոնները ոչ պատշաճ են կառավարվում, ինչի հետևանքով առաջանում են տարբեր խնդիրներ՝

- Թափոնները լցվում են աղբավայրերում, որտեղից քիմիական նյութերը ներծծվում են հողը և ջրային ռեսուրսները,
- Թունավոր գազերի արտանետումներ՝ այրման ընթացքում արտադրվում են վտանգավոր նյութեր, որոնք նպաստում են օդի աղտոտմանը,
- Աշխատուժի շահագործում՝ զարգացող երկրներում հաճախ երեխաներ են ներգրավվում թափոնների մշակման աշխատանքներում, ինչը վնասում է նրանց առողջությունը:

Համաձայն 2023 թվականին անցկացված փորձագիտական ուսումնասիրությանների տվյալների

- Ամեն տարի աշխարհում արտադրվում է մոտ 60 միլիոն տոննա էլեկտրոնային թափոն,
- Դրանց միայն 17-20%-ն է պաշտոնապես վերամշակվում, մնացած մասը նետվում է աղբավայրեր,
- Ամենաշատ թափոններ արտադրող երկրներն են՝ Չինաստանը, ԱՄՆ-ն և Հնդկաստանը:

➢ **Էկոլոգիական անվտանգության ապահովման ուղիներ**

Համակարգչային տեխնիկայի էկոլոգիական ազդեցությունը նվազեցնելու համար անհրաժեշտ են հետևյալ գործողությունները.

- **Վերամշակում և վերօգտագործում**

Տարբեր երկրներում գործում են օրենքներ և կարգավորումներ, որոնք խրախուսում են էլեկտրոնային սարքերի վերամշակումը և թափոնների կառավարումը.

- ԵՄ-ում գործում է WEEE (Waste Electrical and Electronic Equipment) Դիրեկտիվը, որը պարտավորեցնում է արտադրողներին վերամշակել իրենց արտադրած սարքերը,
- ԱՄՆ-ում կան մի շարք օրենքներ, որոնք խրախուսում են վերամշակումը և էկոլոգիապես անվտանգ արտադրությունը,
- Հայաստանում էլեկտրոնային թափոնների կառավարումը դեռևս զարգացման փուլում է, սակայն կան նախաձեռնություններ՝ այս ոլորտը բարելավելու համար:

Հին սարքերի վերամշակումը նվազեցնում է թափոնների ծավալը և թույլ է տալիս օգտագործել երկրորդային հումք: Այս գործընթացը ներառում է պլաստիկի, մետաղների և այլ բաղադրիչների տարանջատումը և վերամշակումը:Համակարգչային տեխնիկայի վերամշակումը, էլեկտրոնային վերամշակումը կամ էլեկտրոնային թափոնների վերամշակումը էլեկտրոնիկայի թափոնների բաղադրիչների և հումքի ապամոնտաժումն ու տարանջատումն է: 2009 թվականին համակարգիչային տեխնիկայի 38%-ը վերամշակվել է Միացյալ Նահանգներում, համապատասխանաբար, 5% և 3% ավելի քան նախորդ 2 տարիներին: Սկսած 1990-ական թվականների սկզբից (ընդհանրապես համակարգչային տեխնիկայի վերամշակման առաջացումից ի վեր) աշխարհում ավելի ու ավելի շատ սարքեր են վերամշակվում իրազեկվածության և ներդրումների ավելացման շնորհիվ: Էլեկտրոնային վերամշակումը տեղի է ունենում առաջին հերթին որպես համակարգչի բաղադրիչ օգտագործված հազվագյուտ և թանկարժեք մետաղները վերականգնելու համար, ինչպես նաև պլաստմասսայի և այլ մետաղների վերամշակման համար: Սրանք վերավաճառվում են կամ մաքրումից հետո օգտագործվում են նոր սարքերում՝ իրականում ստեղծելով շրջանաձև տնտեսություն:

Վերամշակումը համարվում է էկոլոգիապես մաքուր, քանի որ այն կանխում է թափոնների վտանգավոր լինելը, ներառյալ ծանր մետաղների և քաղցկեղածին նյութերի դեպի մթնոլորտ, աղբավայր կամ ջրատարներ մուտք գործելը: Մինչդեռ էլեկտրոնիկան կազմում է առաջացած ընդհանուր թափոնների մի փոքր մասը,

դրանք շատ ավելի վտանգավոր են։ Գոյություն ունեն խիստ օրենսդրական կանոնակարգեր, որոնք նախատեսված են կիրառելու և խրախուսելու համակարգչային սարքերի կայուն վերամշակումը։ Ամենանշանավորները Եվրոպական Միության «Էլեկտրական և էլեկտրոնային սարքավորումների թափոնների մասին հրահանգ»-ն ու Միացյալ Նահանգների «Համակարգչային վերամշակման ազգային ակտ»-ն են։

- ✓ **Կանաչ տեխնոլոգիաների կիրառում**

Արդյունաբերական ընկերությունները պետք է օգտագործեն էկոլոգիապես մաքուր նյութեր և էներգաարդյունավետ տեխնոլոգիաներ: Օրինակ, որոշ արտադրողներ ներդնում են **«կանաչ» միկրոչիպեր**, որոնք նվազեցնում են էներգիայի սպառումը:

- ✓ **Պատասխանատու սպառում**

Օգտատերերը կարող են երկարացնել համակարգիչների շահագործման ժամկետը՝ պարբերաբար սպասարկելով և թարմացնելով ծրագրային ապահովումը: Բացի այդ, կարևորվում է հաշվարկային ամպային տեխնոլոգիաների (cloud computing) կիրառումը, որը նվազեցնում է անհատական սարքերի ծանրաբեռնվածությունը:

- ➢ **Նորարարական մոտեցումներ և ապագայի միտումներ**

Վերջին տարիներին ի հայտ են գալիս նոր տեխնոլոգիաներ, որոնք նվազեցնում են համակարգիչների ազդեցությունը շրջակա միջավայրի վրա.

- **Կենսաքայքայվող նյութեր**՝ որոշ ընկերություններ փորձում են օգտագործել օրգանական նյութեր համակարգիչների պատյանների արտադրության համար,
- **Անհնազանդվող էլեկտրոնիկա** (self-destructing electronics)՝ ստեղծվում են սարքեր, որոնք ծառայության ժամկետից հետո քայքայվում են առանց շրջակա միջավայրի վնասելու:

Այսպիսով, համակարգչային տեխնիկայի արտադրությունը և օգտագործումը էական ազդեցություն ունեն շրջակա միջավայրի վրա, ինչը պահանջում է համապատասխան միջոցառումների իրականացում: Վերամշակման, կանաչ տեխնոլոգիաների և պատասխանատու սպառման խթանումը կարող է զգալիորեն

նվազեցնել համակարգիչների բացասական ազդեցությունը բնության վրա: Ուստի էկոլոգիական անվտանգությունը պետք է լինի ինչպես արտադրողների, այնպես էլ սպառողների առաջնային խնդիրներից մեկը:

Եզրակացություն

Ստեղծվել է համակարգ, որը ինչպես դասախոսներին այնպես էլ ուսուցիչներին հնարավորություն է տալիս գնահատել և հետևել ուսանողների գիտելիքների մակարդակին թեստերի միջոցով: Մշակված համակարգը կարող է ներդրվել ուսումնական հաստատություններում (դպրոցներ, քոլեջներ, բուհեր), խորացված վերապատրաստման դասընթացներում, ինչպես նաև օգտագործվել՝ կորպորատիվ դասընթացներում: Այս հավելվածը նախատեսված է առցանց թեստավորման համար՝ ապահովելով ուսանողների գիտելիքների ստուգման արդար և վերահսկելի միջավայր: Այն հնարավորություն է տալիս դասախոսներին ստեղծել ու անցկացնել թեստեր, իսկ ուսանողներին՝ մասնակցել դրանց:

1. Դասախոսները կարող են ստեղծել թեստեր, սահմանել դրանց պայմանները, անցկացնել և ստանալ ուսանողների արդյունքները:
2. Ուսանողները կարող են մուտք գործել համակարգ, անցնել թեստերը և ստանալ իրենց գնահատականները:
3. Հավելվածը կարող է կիրառվել համալսարաններում, կրթական հաստատություններում և այլ կազմակերպություններում՝ գիտելիքների ստուգման գործընթացի թվայնացման և վերահսկելիության բարձրացման նպատակով:

Համակարգում իրականացված է արդյունքների ավտոմատ գնահատում՝ հիմք ընդունելով ուսանողի կողմից տրված պատասխանները և նախապես սահմանված ճիշտ պատասխանները:

Գրականության ցանկ

1. MDN Web Docs. (2024). React, HTML, CSS, and JavaScript Reference. Retrieved from: https://developer.mozilla.org/
2. W3Schools. (2024). Java Tutorial. Retrieved from: https://www.w3schools.com/java/
3. OWASP Foundation. (2024). OWASP Top Ten Security Risks. Retrieved from: https://owasp.org/www-project-top-ten/
4. •Oracle. (2024). Java SE Documentation. Retrieved from: https://docs.oracle.com/javase/
5. Walls, C. (2022). Spring in Action (6th ed.). Manning Publications.
6. Mak, J. B. (2021). Spring Boot: Up & Running. O'Reilly Media.
7. De Sanctis, V. (2022). Full-Stack Development with Spring Boot and React. Packt Publishing.
8. Banks, A., & Porcello, E. (2023). Learning React (3rd ed.). O'Reilly Media.
9. Younis, S. (2022). Mastering PostgreSQL 15. Packt Publishing.
10. PostgreSQL Global Development Group. (2024). PostgreSQL Documentation. Retrieved from: https://www.postgresql.org/docs/