

ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫң БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
Ш. ЕСЕНОВ атындағы КАСПИЙ ТЕХНОЛОГИЯЛАР ЖӘНЕ ИНЖИНИРИНГ
УНИВЕРСИТЕТІ

Қазақ-Неміс тұрақты инженерия институты

ОҚУ ТӘЖІРИБЕСІ ТУРАЛЫ ЕСЕП

Жоба атавы: Student Information System

Білім беру бағдарламасы: 6B07119 Mechatronics
(БББ ның цифрлық атавы)

Жетекшісі

(көлө)

Орындаған

(көлө)

Орындаған

(көлө)

Орындаған

(көлө)

Орындаған

(көлө)

Орындаған

(көлө)

Орындаған

(көлө)

Кенжебаева Ж.Е.

(Жетекшінің Т.А.Ж)

Оспанов Шахмұрат

(Студенттің Т.А.Ж)

Берікұлы Сержан

(Студенттің Т.А.Ж)

Ұзакберді Жансерік

(Студенттің Т.А.Ж)

Ерғожа Элишер

(Студенттің Т.А.Ж)

Мухтарулы Диас

(Студенттің Т.А.Ж)

Мазмұны

| | |
|--|-----------|
| КІРІСНЕ | 3 |
| Жобаның мақсаты:..... | 3 |
| Негізгі міндеттері:..... | 3 |
| Тақырыптың өзектілігі:..... | 4 |
| БІРІНШІ БӨЛІМ | 5 |
| 1. Сайт аналогтары:..... | 5 |
| 2. Жоба архитектурасы..... | 11 |
| 2.1 Бекенд архитектурасы..... | 11 |
| 2.2 Фронтенд (Клиенттік) Архитектурасы..... | 12 |
| 2.3 Дереккөр архитектурасы..... | 13 |
| Деректердің Тұтастығын Қамтамасыз ету Рөлі..... | 15 |
| ЕКІНШІ БӨЛІМ | 16 |
| 2.1 Бекенд инструменттері (Node.js)..... | 16 |
| 2.2 JavaScript-тің Негізгі Міндеттері..... | 17 |
| 2.3. HTML және CSS..... | 19 |
| 2.4. MySQL..... | 19 |
| 2.5 Қосымша қолданылған кітапханалар..... | 21 |
| ҮШІНШІ БӨЛІМ | 23 |
| 3.1 Сабак кестесін басқару модулі:..... | 23 |
| 3.2 Академиялық календарь модулы..... | 26 |
| 3.3 Бағалау Және Нәтижелер Модулі..... | 28 |
| ҚОРЫТЫНДЫ | 41 |
| 1. Жобаның Нәтижелері және Мақсаттардың Орындалуы..... | 41 |
| 2. Архитектуралық Шешімдердің Тиімділігі..... | 41 |
| 3. Келешектегі Даму Жоспарлары..... | 42 |
| ПАЙДАЛЫНЫЛҒАН ӘДЕБИЕТТЕР | 43 |

KIPICPE

Жобаның мақсаты:

Жобаның негізгі мақсаты – заманауи веб-технологияларды (Node.js, MySQL) қолдана отырып, университет немесе колледж деңгейіндегі шағын студенттік порталдың жұмыс істейтін үлгісін өзірлеу және іске қосу. Бұл портал оку процесінің негізгі ақпараттық қажеттіліктерін қанағаттандыруға бағытталған.

Негізгі міндеттері:

1. Пайдаланушыларды Аутентификациялау және Авторизациялау:
Қауіпсіз тіркелу, кіру және сессияны басқару жүйесін құру. Бұл жүйе әрбір пайдаланушының өзінің рөліне (студент, оқытушы, әкімші) сәйкес келетін ақпаратқа ғана қол жеткізуін қамтамасыз етеді.
2. Деректердің Тұтастығы мен Қауіпсіздігін Қамтамасыз ету:
Парольдерді хештеу және SQL транзакцияларын қолдану арқылы деректерді сақтау мен жаңартудың жоғары сенімділігіне қол жеткізу.
3. Оқу Кестесін Басқарудың Икемді Жүйесін Ұсыну: Студенттер үшін өз тобының кестесін қарауды, ал оқытушылар үшін кестені тиімді түрде жаңартуды (сақтауды) қамтамасыз ету.
4. Курстар Мен Ақпараттық Қолжетімділікті Женілдету: Студенттер үшін олардың тобына бекітілген курстар туралы, ал оқытушылар үшін олардың жүргізетін пәндері туралы мәліметтерді ұсыну.
5. Пайдалануши Интерфейсін Әзірлеу: Әр рөлге арналған интуитивті және функционалды Dashboard (басқару панелі) құру, соның ішінде профильді басқару, парольді ауыстыру және аватарды жүктеу мүмкіндіктерін қосу
6. Мұғалімдер үшін ыңғайлыш, онай бағалау жүйесін қамтамасыз ету.
Автономды семместірлік бағаны қорытындылау. Студент үшін ыңғайлыш баға кестесін көрсетуге мүмкіндік беру.

Тақырыптың өзектілігі:

Қазіргі білім беру жүйесінде цифрландыру және оқу процесінің ақпараттық ашықтығы басты талап болып табылады. Пандемиядан кейінгі кезеңде қашықтықтан және гибридті оқыту форматының өсуіне байланысты, студенттер мен оқытушылардың барлық қажетті ақпаратқа бір терезеден, кез келген уақытта және келген құрылғыдан қол жеткізу мүмкіндігі аса маңызды.

Бұл шағын студенттік портал:

1. Уақытты үнемдейді: Ақпаратты (кесте, курстар) қағаз тасымалдағыштардан немесе көптеген арналардан іздеу қажеттілігін жояды.
2. Басқаруды жеңілдетеді: Оқытушылар мен әкімшілерге кестелер мен курсарды басқаруда тиімді, орталықтандырылған құрал береді
3. Қауіпсіздік артырады: Қазіргі заманғы қауіпсіздік стандарттарын (хештеу, сессияны басқару) қолдану арқылы пайдаланушылардың жеке деректерінің қорғалуын қамтамасыз етеді.

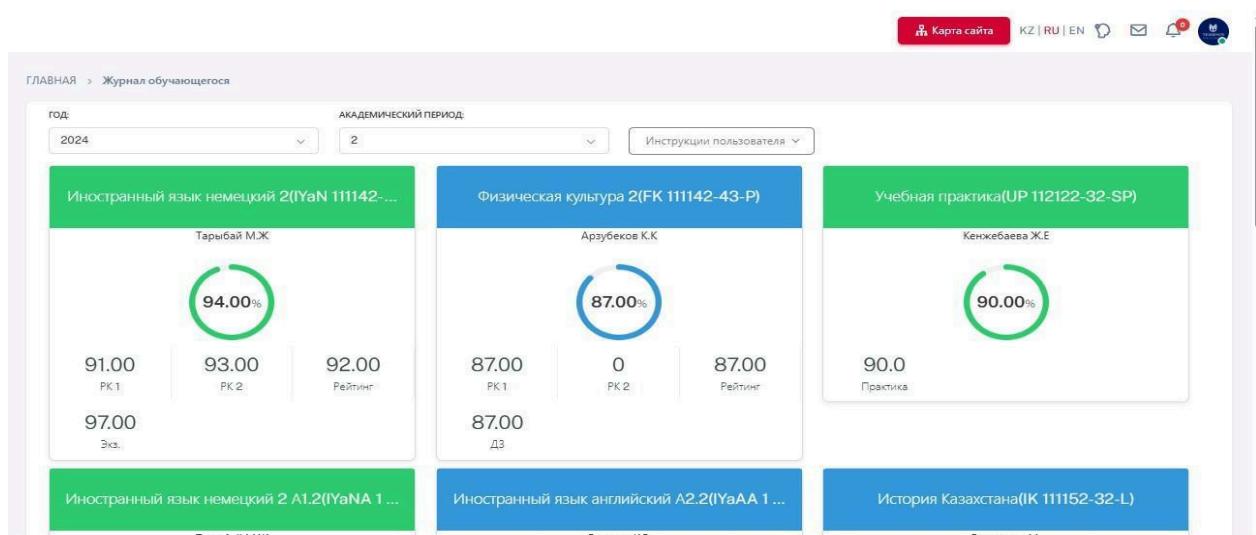
Осылайша, жоба білім беру мекемесінің операциялық тиімділігін арттыруға және оқу процесінің бардық қатысушылары үшін ақпарат алмасудың сапасын жақсартуға тікелей ықпал етеді.

БІРІНШІ БӨЛІМ

1. Сайт аналогтары:

Қазіргі уақытта студенттік порталдар өте көп. Олар ішіндегі кеңінен тарағандары: Moodle, Platonus, Canvas, Univer және т.б. Бұл порталдар жалпы әлемде университеттерде ауқымды қолданылады және оқу процесін қайда жөнделетеді.

Platonus көбінесе Қазақстандағы колледждер мен университеттерде пайдаланылады. Платонус қазақ және орыс тілдеріне толық бейімделген. Білім беру үйымдарының ішкі жүйесіне сай, яғни деканат, бағалар, сабак кестесі, студент деректері, журналдар т.б бәрі бір жерде. Басқаларындағы автоматтандырылған бағалау(1.1 Сурет) және есеп беру жүйесі(1.2 Сурет) жақсы жұмыс жасайды. Және интеграциясы жөндел – ҚР БФМ талаптарына сай есеп беру мүмкіндігі бар (мысалы, академиялық үлгерім т.б).



1.1 - Сурет. Бағалар жүйесі.

| | | | | | | | | | |
|--|----------------------------------|------------|---|------|----|------|---------|---|---|
| 6 | Иностранный язык английский A2.1 | IYaAA 1202 | 4 | 85.0 | B+ | 3.33 | Хорошо | 0 | 0 |
| 7 | Иностранный язык немецкий 1 | IYaN 1106 | 6 | 91.0 | A- | 3.67 | Отлично | 0 | 0 |
| 8 | Политология | Pol 1103 | 2 | 80.0 | B | 3.0 | Хорошо | 0 | 0 |
| Академический период 1 GPA - 3.31 | | | | | | | | | |
| 9 | Физическая культура 2 | FK 1121 | 4 | 87.0 | B+ | 3.33 | Хорошо | 0 | 0 |
| 10 | Физика | Fiz 1222 | 5 | 87.0 | B+ | 3.33 | Хорошо | 0 | 0 |
| 11 | История Казахстана | IK 1114 | 5 | 86.0 | B+ | 3.33 | Хорошо | 0 | 0 |
| 12 | Математика 1 | M 1204 | 5 | 82.0 | B | 3.0 | Хорошо | 0 | 0 |
| 13 | Культурология | Kul 1115 | 2 | 87.0 | B+ | 3.33 | Хорошо | 0 | 0 |
| 14 | Психология | Psi 1116 | 2 | 91.0 | A- | 3.67 | Отлично | 0 | 0 |
| 15 | Иностранный язык английский A2.2 | IYaAA 1203 | 4 | 81.0 | B | 3.0 | Хорошо | 0 | 0 |
| 16 | Иностранный язык немецкий 2 | IYaN 1118 | 4 | 94.0 | A- | 3.67 | Отлично | 0 | 0 |
| 17 | Иностранный язык немецкий 2 A1.2 | IYaNA 1201 | 2 | 94.0 | A- | 3.67 | Отлично | 0 | 0 |
| Академический период 2 GPA - 3.34 | | | | | | | | | |

1 Курс GPA - 3.33 (Минимальный GPA для перевода с курса на курс : 0)

©Platonus v6.25.9 (build# 15), 2006-2025 Platonus

1.2 - Сурет. Транскрипт.

Canvas әлемдік деңгейде кең таралған, мысалға айтсақ АҚШ, Еуропадағы университеттерде. Канвас интерфейсі айтарлықтай заманауи және ынғайлы навигациямен жабдықталған(1.3Сурет). Интерактивті құралдар көп, мәселен форумдар, тест, бейнежазбалар, тапсырма беру, кері байланыс(1.4 Сурет) және ынғайлы бағалау системасы (1.5 Сурет). Бұлттық сақтау жүйелерімен (Google Drive, OneDrive) оңай байланысады. Аналитика және статистика арқылы студенттің белсенділігін көруге болады. Алайда Canvas интернетке тәуелді, яғни онлайн режимде жақсы жұмыс істейді. Және бір кемшілігі толық функционалды нұсқасы ақылы, бірақта бұл университетке қыын емес деп санаймыз. Және интерфейс кейде ағылшын тілінде болғандықтар, басқа тілге бейімделу уақытын қажет етеді.

The screenshot shows a dashboard with a sidebar on the left containing icons for Account, Dashboard, Courses, Groups, Calendar, Inbox, Commons, Studio, and Help. The main area displays several course tiles:

- Physical Science**: Physical Science | 2019-2020, Physical Science
- MATH**: Grade 8 Math, featuring PEMDAS and mathematical formulas.
- First Grade**: First Grade, First
- Teaching Digital Citizenship**: Digital Citizenship - Student DigCit
- Mahwah Curriculum Hub**: MCH, featuring a circular diagram with 'UNIVERSAL DESIGN FOR LEARNING' in the center.
- SEL**: Social-Emotional Learning SEL

To the right, there is a 'To Do' list with 16 items, each with a small thumbnail and a delete icon:

- Grade Experimental Design Feedback: Hypothesis
- Grade Nonfiction Main Idea: Individual Project
- Grade Review: The Scientific Method - Quiz
- Grade Scientific Investigation: Bell Ringer
- Grade Experimental Design Feedback: Hypothesis

Below the list is a video call interface showing a man's face.

1.3 - Сурет. Canvas басты беті.

The screenshot shows a monthly calendar for March 2020. The sidebar on the left is identical to the one in the previous screenshot. The calendar grid shows various tasks and assignments for each day:

| SUN | MON | TUE | WED | THU | FRI | SAT |
|-----|--|--|----------------------|---|--|---------------------------|
| | | | | | | |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| | Scientific Investigation: Bell Ringer | Spanish HW 13 | MATH HW 2.3 | New Assignment Example | Experimental Design Feedback: Hypothesis | MIDTERM |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| | Name | Separating Mixtures and Solutions - Quiz | Lesson Plan Skeleton | Experiment Approval Assignment Submission | MIDTERM REVIEW | Name (PHYSICI - Period 1) |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| | Experiment Approval Assignment Submission (PHYSICI - Period 1) | Math HW 2.2 | MATH 2.4 | MATH 2.5 | Microsoft Teams Sync | |
| 29 | 30 | 31 | 1 | 2 | 3 | 4 |
| | Experiment Approval Assignment Submission | Microsoft Teams Sync | | | | |

On the right side, there is a legend for calendars and an 'UNDATED' section with a video call interface.

1.4 - Сурет. Календарь жүйесі

The screenshot shows a Moodle mobile application interface. At the top, it displays the time (3:30), signal strength, battery level (65%), and course name (AP Physics 1 C (Stern Q3 ...)). Below this, there are two tabs: 'GRADES' and 'FRONT PAGE'. The 'GRADES' tab is selected.

Total Grade: B+ (with a red arrow pointing to it)

Assignments:

- homework for 1/31: Due Jan 31 at 11:59 PM. Not Submitted. Score: 1 / 1
- homework for 2/8: Due Feb 8 at 11:20 AM Late. Score: 1 / 1
- homework for 2/20: Due Feb 20 at 8:30 AM Late. Score: 1 / 1
- Practice: Gravitation Review: Due Feb 23 at 9:46 AM Submitted. Score: 1.5 / 2

tests:

- rotational kinematics and torques test: No Due Date. Score: 90 / 100
- Cool Things in Space for Fun and Enrichment: Due Feb 23 at 11:59 PM Late. Score: 0 / 0 (indicated by a red circle with a speech bubble icon)

Total: 89.81%

| Name | Due | Submitted | Status | Score |
|--|-------------------|-------------------|--------|---------------|
| homework for 1/31 Assignments | Jan 31 by 11:59pm | | | 1 / 1 |
| homework for 2/8 Assignments | Feb 8 by 8:20am | Feb 14 at 8:05pm | late | 1 / 1 |
| homework for 2/20 Assignments | Feb 20 by 8:30am | Feb 26 at 4:46pm | late | 1 / 1 |
| Practice: Gravitation Review Assignments | Feb 23 by 9:46am | Feb 23 at 9:05am | | 1.5 / 2 |
| Cool Things in Space for Fun and Enrichment Tests | Feb 23 by 11:59pm | Feb 24 at 6:28pm | late | 5 / 5 |
| Practice: NLUG Reasoning Imported Assignments | Feb 29 by 11:59pm | Feb 29 at 10:52pm | | 0 / 0 |
| Harmonic Motion Project Tests | Mar 11 by 11:59pm | Mar 11 at 8pm | | 49 / 50 |
| angular momentum and rotational energy Tests | | | | 37.5 / 50 |
| Practice: Newton's 2nd Law for Rotation Imported Assignments | | | | 0 / 0 |
| rotational kinematics and torques test Tests | | | | 90 / 100 |
| Assignments | | | | 100% |
| Tests | | | | 88.54% |
| Final | | | | N/A |
| Imported Assignments | | | | N/A |
| Total: | | | | 89.81% |

1.5 - Сурет. Бағалар тақтасы

Moodle әлем бойынша ең көп қолданылатын ашық кодты (open-source) LMS. Бұл портал толығымен тегін және икемді. Кез келген универ өзіне ыңғайлы, қажет етіп жекешелендіре алады (плагиндер, дизайн, модульдер). Сонымен қатар көп функциялы: форум, тест, чат, файл жүктеу, бағалау нұсқаулықтар, сияқты. Айтақандай мұғалім үшін порталда көптеген функциялар жұмыс жасайды, яғни жеке курсымен жұмыс жасаудан(1.6 Сурет) бастап онлайн сабактар жүргізіп, кері байланыс алуға дейін(1.8 Сурет). Бұл өз кезегінле Және бұл портал онлайн және оффлайн жұмыс жасау мүмкіндігі бар. Бірақ кейде интерфейсі ескі және күрделі көрінеді. Және басқада порталдар секілді әрине бұл порталда сервер баптауы мен техникалық қызметі арнайы маманды қажет етеді. Тағы бір айтатын нәрсе жоғарыда айтылғандай көп плагин орнатылса жүйе баяулайды.

Object Oriented Programming Java (Akizhanov Mugzar)

Home / Courses / 2019-2020 / ICT / 3rd semester / Java-osp-2019-03-term



6 April - 12 April



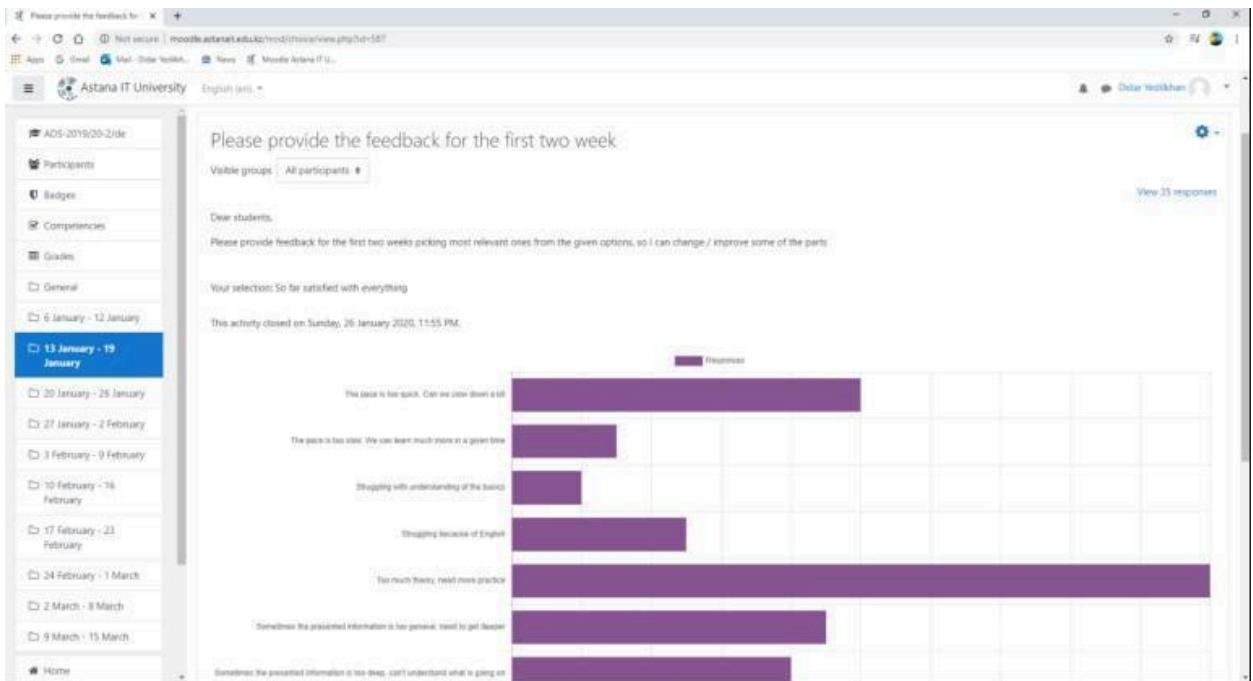
you can get free student license by filling this application



1.6 - Сурет.Мұғалім үшін курсы беті

A screenshot of a Moodle assignment grading page titled "Assignment 1". The page shows a table of student submissions. One submission from user "Irsen Altynzha" is highlighted. The table columns include User picture, First name / Surname, Email address, Status, Grade, Last modified (submission), File submissions, Submission comments, Last modified (grade), Feedback comments, and Final grade. The status for the highlighted submission is "Submitted for grading Graded". The grade is 100.00 / 100.00. The submission date is Saturday, 11 April 2020, 9:38 PM. The file submissions listed are AssignmentOne.java, print.java, and shape.java. The submission comments show a comment from the teacher dated Wednesday, 29 April 2020, 12:50 PM. The final grade is 100.00 / 100.00. On the left, a sidebar lists assignment periods: 6 April - 12 April (selected), 13 April - 19 April, 20 April - 26 April, 27 April - 3 May, 4 May - 10 May, 11 May - 17 May, 18 May - 24 May, 25 May - 31 May, 1 June - 7 June, 8 June - 14 June, Home, and Dashboard.

1.7 - Сурет.Оқушыларды бағалау мен бақылау



1.8 - Сурет. Откен сабак туралы анализ

| Критерий | Platonus | Canvas | Moodle |
|--------------------------|----------------|--------------------------|-------------------------|
| Тілі | Қазақша\орысша | Ағылшын(кейде көп тілді) | Көптілді |
| Қолайлылық | Орташа | Өте жоғары | Орташа |
| Бағасы | Лицензиялы | Ақылы | Тегін |
| Көрінісі(UI) | Ескі | Заманауи | Классикалық н\е жеке |
| Техникалық баптау | Оңай | Дайын жүйе | Өз бетімен баптау қажет |
| Интерактив | Тәмен | Жоғары | Орташа\Жоғары |

1.1- Кесте. Аналогтар анализі

Осы сайттарды назарға ала отыра (1-Кесте) бізде сайтымызға бардық студенттік порталдар секілді сабак кестесі, бағалар, және курстарды қосамыз.

Осы аналогтардағы студент пен мұғалім жұмысын жеңілдедетін динамикалармен функциялармен проектті дамытамыз.

2. Жоба архитектурасы.

2.1 Бэкенд архитектурасы.

Жоба Backend-as-a-Service (BaaS) архитектурасын қолданады, мұнда Node.js-тегі сервер API-сервері және статикалық файлдар сервері ретінде жұмыс істейді.

Бэкенд логикалық түрғыдан екі негізгі модульге бөлінген:

1. Сервер (server.js): HTTP-сұрауларды маршруттауға, статикалық файлдарды беруге және бизнес-логиканы шақырmas бұрын авторизация мен рөлдерді алдын ала тексеруге жауап береді.
2. Авторизация және Деректер Модулі (auth.js): Барлық бизнес-логиканы және дерекқормен тікелей өзара әрекеттесуді қамтиды.

Бекендтің басты міндеті – рөлдік модельді қатаң бақылау (users.role).

- Қолжетімділікті фильтрлеу: Серверге келіп түскен әрбір сұраныс checkSession функциясы арқылы тексеріледі.
- Рұқсаттық Тексеру: server.js ішіндегі маршруттық логика пайдаланушының белгілі бір әрекетті орындауға (мысалы, saveSchedule - тек оқытушиға рұқсат) рұқсаты бар-жоғын тексереді. Бұл фронтендтегі кез келген айла-шарғыдан қорғаныстың соңғы шебі болып табылады.

Бекенд барлық деректер операцияларын тиімді басқарады:

- Дерекқормен Байланыс: Қосылу пулы арқылы дерекқорға сенімді және өнімді қатынауды қамтамасыз етеді (db.js).

- Транзакциялық Басқару: Кесте сияқты маңызды деректерді жаңарту кезінде (`saveSchedule`), деректердің тұтастығын сақтау үшін SQL транзакцияларын қолданады. Бұл сервердің кез келген қателік кезінде деректерді бастапқы күйіне қайтаруына кепілдік береді.

2.2 Фронтенд (Клиенттік) Архитектурасы.

Архитектуралық Үлгі: Динамикалық Бірбеттік Қосымша (SPA элементтері бар). Жоба таза SPA (Single Page Application) болмаса да, ол SPA элементтерін қолданады. Себебі:

1. Негізгі интерфейс Dashboard (dashboard.html немесе оған ұқсас) бір рет жүктеледі.
2. Содан кейін барлық ақпарат (кесте, курстар, профиль деректері) AJAX-сұраулары (асинхронды API шақырулары) арқылы жүктеледі.
3. Пайдаланущы интерфейсі бетті толық қайта жүктемей, тек JavaScript арқылы динамикалық түрде жаңартылады.

Фронтенд архитектурасының басты ерекшелігі – оның API-ға бағытталуы:

- Асинхронды Деректер Алмасу: Интерфейстің көп бөлігі бетті толық қайта жүктемей, тек API маршруттарына (мысалы, /api/get_schedule) сұрау жіберу арқылы жаңартылады. Бұл пайдаланущы тәжірибесін жылдам әрі үздіксіз етеді.
- Деректерді Визуализациялау: Бекендтен JSON форматында алғынған деректер Интерактивті Логика арқылы өнделеді де, оқуға оңай HTML кестелеріне немесе тізімдерге айналып, Құрылымдық Қабаттағы тиісті орындарға орналастырылады.

Бекендтен алғынған пайдаланушының рөліне (student, teacher, admin) сәйкес, фронтенд интерфейсті адаптациялайды:

- Қажетті Элементтерді Көрсету: Тек рұқсат етілген элементтерді көрсетеді (мысалы, Админ ғана КҮРСТЫ ҚОСУ формасын көрсету).
- Функционалды Шектеу: Белгілі бір рөлдер үшін түймелерді немесе форма өрістерін өшіру (disable) арқылы рұқсат етілмеген әрекеттерді болдырмайды.

Қорытынды: Фронтенд архитектурасы жеңіл, динамикалық және API-ға бағытталған (API-Driven) болып табылады. Ол берілген ролдік модельді сақтай отырып, бекендтен алғынған деректерді тиімді көрсетеді, және JavaScript арқылы DOM-ды тікелей басқаруға негізделген.

2.3 Дерекқор архитектурасы.

Реляциялық моделі.

Архитектураның негізі — Реляциялық Модель, мұнда деректер бірнеше логикалық кестелерге бөлінген, олар бір-бірімен айқын байланыстар арқылы біріктірілген.

- Users Кестесі: Барлық пайдаланушылар туралы деректердің орталық тізімі. Бұл кесте басқа барлық кестелер үшін бастапқы кілт (Primary Key) ретінде қызмет етеді.
- Sessions Кестесі: Қауіпсіздік үшін қолданылады. Users кестесімен байланысы бар, авторизацияланған пайдаланушы күйін және оның жарамдылық мерзімін бақылайды.
- Schedule & Courses Кестелері: Негізгі оқу ақпаратын сақтайды. Users кестесімен сыртқы кілттер (Foreign Keys) арқылы байланысады, бұл кестелердегі мұғалімнің кім екенін анықтауға мүмкіндік береді.
- admin_messages кестесі: Студент жағынан админге жазылатын хабарламаларды сақтайды. (is_read ? - логикасы)
- students_grade: Студент үшін Результаты бөліміндегі OverViewResults тақтасының мәліметтің сақтайды.

- academic_events: Басты беттегі Академиялық календарь мәліметтерін сақтайды.
- assessment_types: Мұғалім үшін баға қою кестесінің шаблондарын сақтайды.
- student_assessments: Студенттердің негізгі бағаларын сақтайды.

| Кесте | Негізгі рөл | Негізгі бағандар |
|-----------------|---|---|
| users | Пайдаланушыларды сақтау.(Студент, оқытушы, Админ) | -id, -username, -password(hashed), -role, -group_name, -full_name, (-avatar_path) |
| session | Пайдаланушының авторизацияланған күйін сақтау. | -id, -user_id, -session_id, -expires_at |
| schedule | Сабак кестесі туралы мәліметтер. | -id, -teacher_id, -group_name, -day_of_week, -time_slot, -course, -classroom. |
| courses | Оқу курсарының тізімі | -id, -name, -description, -credits, -teacher_id, -group_name, -semester. |
| student_grades | OverViewResults тақташасының мәліметтері | -id, -user_id, -course_id, -final_score, -letter_grade, -semester_code |
| academic_events | Академиялық күнтізбе мәліметтерін сақтайды | -id, -title, -start_date, -end_date, -description, -color, -created_at |

| | | |
|---------------------|--|---|
| assessment_types | Мұғалімдер үшін баға қою кестесін шаблондармен қамтамасыз етеді. | -id, -course_id, -assessment_name, -category, -subcategory, -weight, -max_score |
| student_assessments | Студенттердің әр пән бойынша бағаларын сақтайды | -id, -student_id, -course_id, -assessment_type_id, -score, -date_recorded |

1.2- Кесте. Мәліметтер базасының құрылымы

Деректердің Тұтастығын Қамтамасыз ету Рөлі

БД архитектурасы деректердің дұрыстығы мен сенімділігін келесі жолдармен қорғайды:

- Транзакцияларды Қолдау: Кестені жаңарту сияқты күрделі операциялар үшін Транзакциялар механизміне қолдау көрсетеді. Бұл екі немесе одан да көп әрекетті атомарлы (бүтін) бір операция ретінде орындауға мүмкіндік береді – не бәрі сәтті өтеді (COMMIT), не ештеңе өзгермейді (ROLLBACK).
- Кілттер мен Индекстер: Бастапқы және Сыртқы кілттерді пайдалану арқылы деректердің логикалық байланысының бұзылмауын қамтамасыз етеді. Бұл сондай-ақ іздеу операцияларының жылдам орындалуына мүмкіндік береді.

Корытынды: Жобаның дерекқор архитектурасы қарапайым, реляциялық принциптерге негізделген және функционалдық қажеттіліктерді толық қанағаттандырады. users, sessions, schedule, courses, student_grades, academic_events, assessment_type және student_assessment кестелерінің айқын бөлінуі мен тиісті байланыстары қосымша қолдау көрсетуді және көбейтуді жеңілдетеді. Дерекқор pool - ын және транзакцияларды қолдану жүйенің сенімділігі мен өнімділігін арттырады.

ЕКІНШІ БӨЛІМ

2.1 Бэкенд инструменттері (Node.js).

Неліктен Node.js ?

- Бірыңғай Тілдік Кеңістік (Full-Stack JavaScript). Ең маңызды фактор — бекенд (Node.js) және фронтенд (браузер) үшін тек бір тілді, яғни JavaScript-ті қолдану мүмкіндігі. Бұл бастаушы әзірлеуші үшін немесе шағын команда үшін оқып-үйренуді, код жазуды және қолдауды айтартықтай жеңілдетеді. Және де әзірлеушіге екі түрлі тілдің (мысалы, Python/PHP және JavaScript) синтаксисін, құрылымын және әртүрлі кітапханаларын меңгеру қажет емес.
- Жаңадан бастаушылар үшін. JavaScript бастапқы бағдарламалау тілі ретінде кең таралған. Node.js-тің өзі конфигурациялауы жеңіл және үлкен, күрделі фреймворктерді қажет етпейді (біздің жобадағыдай, тіпті "Vanilla" Node.js-ті қолдану). Бұл платформада қарапайым HTTP API-ды тез құруға болады, бұл біздің студенттік портал сияқты шағын және орташа жүйелерге өте қолайлы.
- Жоғары өнімділік (Асинхрондылық). Node.js асинхронды және оқиғаға негізделген (event-driven) архитектурада жұмыс істейді. Бұл дегеніміз, ол бір уақытта көптеген сұраныстарды тиімді өндей алады. Дерекқорға немесе файлдарға сұрау салу сияқты ұзақ операциялар кезінде, [Node.js](#) бұл процесті аяқтауды күтпейді, керісінше, басқа сұраныстарды өндеуді жалғастырады. Біздің жобадағыдай SQL сұраулары сияқты I/O (енгізу/шығару) операциялары көп болатын қосымшалар үшін бұл өте тиімді.
- Қарапайымдылық. Node.js-тің әлемдегі ең үлкен пакет менеджері NPM (Node Package Manager) бар. Бұл дегеніміз, қажетті функцияны іске асыру үшін дайын шешімдер (мысалы, crypto немесе formidable) оңай және тез қол жетімді. Бұл кітапханалардың молдығы жаңа

функционалды (мысалы, аватар жүктеуді) жылдам енгізуге мүмкіндік береді, жобаны дамыту уақытын қысқартады.

Нодада (Node.js) сұрауларды (запростарды) қабылдау және өндөу әдетте роутинг (маршруттау) арқылы жүзеге асады. Сіздің жобаңызыдағы server.js файлы дәл осыны істейді. Ол клиенттен келген HTTP сұрауын (GET, POST) қабылдайды, оның жолын (URL) анықтайды және тиісті функцияны шақырады.

```
if(req.method === "POST" && pathname === "/register") {
    let body = "";
    req.on("data", chunk => body += chunk);
    req.on("end", async () => {
        const data = JSON.parse(body);
        const result = await register(data.username, data.password, data.role);
        res.end(JSON.stringify(result));
        return;
    });
    return;
}
```

2.1-Сурет. Сервердің POST сұрауы

Бұл код үзіндісі (2.1 - Сурет) Node.js HTTP серверінің клиенттен келген POST сұрауын қалай өндейтінін көрсетеді: ол алдымен '/register' жолын сүзеді (if(req.method === "POST" && pathname === "/register")), содан кейін req.on("data", ...) арқылы деректерді асинхронды жинап алады, req.on("end", ...) ішінде жиналған деректі JSON.parse() арқылы талдайды, register(...) бизнес-логикасын шақырып, res.end(...) арқылы тіркеу нәтижесін клиентке JSON форматында қайтарады.

2.2. JavaScript-тің Негізгі Міндеттері.

Фронтенд JavaScript коды келесі маңызды міндеттерді орындаиды:

А. Авторизацияны Басқару және Рөлге Қатысты Рендеринг

- Сессияны Тексеру: Бастапқы жүктеу кезінде (немесе onload ішінде) клиенттік JS бекендтегі /api/check_session сияқты маршрутқа сұрау жібереді.
- Интерфейсті Бейімдеу: Бекенд қайтарған рөлге (student, teacher, admin) негізделген.
 1. Егер рөл "student" болса, тек қарауға арналған элементтерді көрсетеді.
 2. Егер рөл "teacher" болса, "Кестені сақтау" түймесін немесе курсарды редакциялау формаларын белсендіреді.
 3. Егер рөл "admin" болса, "Курстарды Басқару" толық формасын көрсетеді.

Б. Деректерді Асинхронды Жүктеу және Көрсету (Data Fetching & Rendering)

- API-мен Өзара Әрекеттесу: fetch() API немесе XMLHttpRequest сияқты құралдар арқылы бекендке сұраулар жібереді. Мысалы:
 1. Кесте: Пайдаланушы "Сейсенбі" күнін таңдағанда, /api/get_schedule маршрутына топ атауымен сұраныс жіберіледі.
 2. Курстар: Студент үшін /api/get_student_courses сұранысы жіберіледі.
- Динамикалық Рендеринг (DOM Manipulation): Бекендтен алғынған JSON деректері (мысалы, кесте сабактарының массиві) қабылданып, JavaScript арқылы HTML-элементтерге (кесте жолдарына, тізімдерге) айналдырылады және тиісті контейнерлерге қойылады.

В. Формаларды Басқару және Деректерді Жіберу

- Кестені Сақтау Формасы: Оқытушының кестені өзгерту формасынан деректерді жинаиды.
- API-ға Жіберу: Деректерді жинақтап, оны JSON.stringify() арқылы форматтап, POST сұрауымен /api/save_schedule маршрутына жібереді.
- Пайдаланушыға Хабарлау: Бекендтен қайтарылған нәтижені ({success:

true} немесе {success: false, error: ...}) алып, экранға хабарлама (уведомление) шығарады.

2.3. HTML және CSS.

HTML (dashboard.html және т.б) мазмұндық қаңқаны және өзара әрекеттесу элементтерін (кнопкалар, формалар, кестелер) анықтайды. Порталдың негізгі сүйегі. Ол тек статикалық құрылымды береді; барлық динамикалық деректер (кестелер, курстар тізімі) JavaScript арқылы осы HTML - элементтерге қойылады.

Визуалды Дизайнды және Орналасуды (Layout) басқарады. Интерфейстің сыртқы көрінісін, түстерін, қаріпперін, өлшемдерін және әсіресе мобиЛЬДІ құрылғыларға бейімделуін (адаптивті дизайн) қамтамасыз етеді. CSS деректердің көрсетуді функционалды және эстетикалық түрғыдан тиімді етеді.

2.4. MySQL

деректер базасының кең таралған басқару жүйесі (СУБД). Сіздің жобаңыз үшін MySQL-ді тандаудың бірнеше негізгі артықшылықтары (плюстары) бар:

1. Тұтастыққа Кепілдік: MySQL реляциялық дерекқор болғандықтан, ол деректердің тұтастығын (целостность) қамтамасыз етеді. Бұл дегеніміз, кестелер арасындағы байланыстар (Foreign Keys) мен транзакциялар (мысалы, кестені сақтау кезіндегі saveSchedule функциясындағыдей) деректердің логикалық сәйкесіздікке ұшырамауына кепілдік береді.
2. Оңай Менгеру: MySQL SQL (Structured Query Language) тілін пайдаланады, бұл тіл оқуға және түсінуге оңай. Бастапқы деңгейдегі өзірлеушілер үшін бұл тілмен жұмыс істеу қыындық тудырмайды. Бағдарламалық жасақтаманы орнату және конфигурациялау процесі қарапайым. MySQL-дің көптеген нұсқалары (Community Edition) тегін қолжетімді, бұл жоба бюджеті шектеулі болғанда өте тиімді.

3. Өнімділік (Performance): MySQL кіші және орташа өлшемді жүктемелер үшін (біздің шағын портал сияқты) өте жоғары жылдамдық пен өнімділік көрсетеді. Пулдармен Тиімділік: Node.js-те қосылым пулымен (pool) бірге қолданылған кезде, MySQL сұрауларды жылдам өндейді, бұл жүйенің жалпы жауап беру уақытын жақсартады.

Дерекқормен өзара әрекеттесу db.js файлы арқылы жүзеге асырылады. Бұл тәсіл әдеттегідей сенімді және тиімді болып табылады:

Технология.

Node.js үшін арналған mysql2/promise драйвері. promise (уәде) функционалын қолдану асинхронды операцияларды (SQL сұрауларын) оңай және таза өндеуге мүмкіндік береді.

Қосылу Механизмі: Пул (Pool). Әрбір API сұрауы үшін дерекқорға қайта-қайта қосылудың орнына, пул алдын ала белгіленген қосылымдар жиынтығын басқарады. Бұл сервердің өнімділігін және көптеген пайдаланушы бір мезгілде кірген кездегі тұрақтылығын айтарлықтай арттырады. Конфигурация: Қосылу жергілікті серверге (localhost немесе 127.0.1.16) стандартты MySQL порты (3306) арқылы "db_01" базасына жасалған.(2.2-Сурет)

```
const mysql = require("mysql2/promise");

const pool = mysql.createPool({
    host: "127.0.1.16",
    user: "root",
    password: "",
    database: "db_01",
    port: 3306
});

module.exports = pool;
```

2.2-Сурет. Сервер мен мәліметтер базасының байланысы

2.5 Қосымша қолданылған кітапханалар.

1. CRYPTO Модулі (Node.js Ишкі Модулі)

crypto – бұл Node.js-тің криптографиялық функцияларға қол жеткізуді қамтамасыз ететін негізгі модуль. Ол хештеу, шифрлау және кездейсок байттарды генерациялау үшін қолданылады.

- Парольді Хештеу (SHA256): hashPassword функциясы (auth.js).
crypto.createHash("sha256").update(password).digest("hex")
арқылы пайдаланушы парольдерін бір бағытты хештейді.(2.3 – Сурет)

```
function hashPassword(password) {  
    return crypto.createHash("sha256").update(password).digest("hex");  
}
```

2.3 – Сурет. Құпия сөзді хештеу

2. MYSQL2/PROMISE Модулі (NPM Пакеті)

Бұл – Node.js қосымшаларын MySQL дерекқорымен байланыстыруға арналған тиімді драйвер. Оның "promise" бөлігі асинхронды операцияларды (SQL сұрауларын) оңай басқаруға мүмкіндік береді.

Асинхронды Сұраулар: await pool.query(...) немесе await pool.execute(...) арқылы SQL сұрауларын орындаиды. Дерекқор операциялары кезінде сервердің блокталмай жұмыс істеуін және кодты async/await арқылы оқуға жеңіл ету.

3. FORMIDABLE Модулі (NPM Пакеті)

formidable – бұл Node.js-тегі файлдарды жүктеуді (upload) және формаларды өндеуді, әсіреле multipart/form-data типті сұрауларды жеңілдететін күшті модуль.

Аватар Жүктеуді Өндөу:

- const form = new formidable.IncomingForm(); form.parse(req, ...) арқылы клиент жіберген файлды қабылдайды. Пайдаланушының аватар файлын серверге тиімді тұрде жүктеп, сақтау. Модуль файлдың өлшемі мен жолын автоматты анықтайды.(2.4 – Сурет)

```
const form = new Formidable({
  uploadDir: UPLOAD_DIR,
  keepExtensions: true,
  maxFileSize: 5 * 1024 * 1024, // 5 МБ
});

form.parse(req, async (err, fields, files) => {
  if (err) {
    console.error("Formidable error:", err);
    res.writeHead(400);
    res.end(JSON.stringify({ success: false, error: "Ошибка при разборе файла." }));
    return;
  }
})
```

2.4 - Сурет.new formidable.IncomingForm() кітапханасы

4. HTTP Cookies (Стандартты Веб Механизмі)

Cookies – бұл сервер клиенттің браузерінде сақтау үшін жіберетін кішкентай деректер бөліктері.

ҮШІНШІ БӨЛІМ

Автоматтандырылған жүйенің негізгі функционалдық модульдері:

3.1 Сабак кестесін басқару модулі:

Бұл модуль Студенттер мен Оқытушылардың ақпарат алу қажеттіліктерін, сондай-ақ Әкімшілік пен Оқытушылардың кесте құру/өндөу міндеттерін шешеді.

| Рөл | Негізгі әрекет | Техникалық іске асыру |
|-----------------|---|--|
| Студент | Өзінің группасына арналған кестені көреді | dashboard.html арқылы student_schedule.html жүктеледі. fetch сұранысы user.group_name негізінде /api/get_schedule маршрутына жібереді.(3.1 - 3.2 Суреттер) |
| Оқытушы \ Админ | Топты және күнді таңдау арқылы көз келген кестені көреді, өндейді және сақтайды | teacher_schedule.html интерфейсі қолданылады. Деректер loadTeacherSchedule() функциясы арқылы жүктеледі, ал сақтау saveSchedule() функциясы арқылы /api/save_schedule маршруты арқылы жүзеге асады. (3.3 - 3.4 Сурет) |

3.1- Кесте. Сабак кестесін басқару модулі

```

else if(req.method === "GET" && pathname === "/api/get_schedule") {
    const cookies = cookie.parse(req.headers.cookie || '');
    const sessionId = cookies.session;
    const user = await checkSession(sessionId);

    if (!user) {
        res.writeHead(401);
        res.end(JSON.stringify({success: false, error: "Неавторизованный доступ."}));
        return;
    }

    //Определение группы
    let groupName;
    let dayOfWeek = parsedUrl.query.day;

    if(user.role === 'student') {
        groupName = user.group_name;
    } else if(user.role === 'teacher' || user.role === 'admin') {
        // Учитель может рассматривать любую группу, которую он выберет.
        // Для простоты, здесь мы будем запрашивать группу из URL-параметров.
        // /api/get_schedule?group=Логистика
        groupName = parsedUrl.query.group;
    } else {
        res.writeHead(403);
        res.end(JSON.stringify({success: false, error: "Нет прав для просмотра расписания"}));
        return;
    }

    // Получение данных
    const scheduleData = await getSchedule(groupName, dayOfWeek);

    if(scheduleData) {
        res.writeHead(200, {"Content-Type": "application/json"});
        res.end(JSON.stringify({success: true, schedule: scheduleData, group: groupName}));
        return;
    } else{
        res.writeHead(500);
        res.end(JSON.stringify({success: false, error: "Ошибка при получении данных расписания."}));
        return;
    }
}

```

3.1 Сурет. Серверлік сұрау (Запрос)

```

async function getSchedule(groupName, dayOfWeek = null) {
    if (!groupName) return [];
    try {
        let query = `

        SELECT
            s.*,
            u.username AS teacher_name
        FROM
            schedule s
        JOIN
            users u ON s.teacher_id = u.id
        WHERE
            s.group_name = ?`;
        let params = [groupName];

        // ДОБАВЛЯЕМ ФИЛЬТР ПО ДНЮ, ЕСЛИ ОН ПЕРЕДАН
        if (dayOfWeek) {
            query += ` AND s.day_of_week = ?`;
            params.push(dayOfWeek);
        }

        query += `

        ORDER BY
            FIELD(s.day_of_week, 'Понедельник', 'Вторник', 'Среда', 'Четверг', 'Пятница'),
            s.time_slot
        `;

        const [rows] = await pool.query(query, params); // <-- ИСПОЛЬЗУЕМ НОВЫЙ МАССИВ ПАРАМЕТРОВ

        return rows;
    } catch (error) {
        console.error("Error fetching schedule:", error);
        return null;
    }
}

```

3.2 Сурет. Деректерді алу\өндөу коды

```
async function loadTeacherSchedule() {
    const group = document.getElementById('groupSelector').value;
    const day = document.getElementById('daySelector').value;
    const msgBox = document.getElementById('scheduleMessage');

    const tableBody = document.getElementById('scheduleInputTable').querySelector('tbody');
    tableBody.querySelectorAll('input').forEach(input => input.value = '');

    if (!group || !day) {
        msgBox.textContent = 'Выберите группу и день недели для загрузки.';
        msgBox.style.color = 'blue';
        return;
    }

    msgBox.textContent = `☒ Загрузка расписания для ${group} (${day})...`;
    msgBox.style.color = 'orange';

    try {
        // Отправляем GET-запрос с выбранными параметрами
        // Мы используем /api/get_schedule?group=X, как вы настроили в server.js
        const res = await fetch(`'/api/get_schedule?group=${group}&day=${day}`);
        const data = await res.json();

        if (data.success && data.schedule && data.schedule.length > 0) {
            // Заполняем форму данными (schedule - это массив уроков)
            data.schedule.forEach(lesson => {
                const row = tableBody.querySelector(`tr[data-time="${lesson.time_slot}"]`);
                if (row) {
                    // Заполняем предмет и аудиторию
                    row.querySelector('input[data-field="course"]').value = lesson.course;
                    row.querySelector('input[data-field="classroom"]').value = lesson.classroom;
                    // Преподавателя заполнять не нужно, т.к. его ID берется из сессии при сохранении
                }
            });
            msgBox.textContent = `☒ Расписание для ${group} (${day}) загружено. Можно редактировать.`;
            msgBox.style.color = 'green';
        } else {
            msgBox.textContent = `⚠️ Расписание для ${group} (${day}) не найдено. Можно составить новое.`;
            msgBox.style.color = 'orange';
        }
    }
}
```

3.3 Сурет. Кесте бетін жүктөу

```

async function saveSchedule(teacherId, groupName, dayOfWeek, lessons) {
    let connection;
    try{
        connection = await pool.getConnection();
        await connection.beginTransaction(); //Начинаем транзакцию для безопасности

        // УДАЛЯЕМ старые записи для этой комбинаций ГРУППА + ДЕНЬ
        await connection.query(
            'DELETE FROM schedule WHERE group_name = ? AND day_of_week = ?',
            [groupName, dayOfWeek]
        );

        // ВСТАВЛЯЕМ новые записи
        if (lessons && lessons.length > 0) {
            const values = lessons.map(lesson => [
                teacherId,
                groupName,
                dayOfWeek,
                lesson.time_slot,
                lesson.course,
                lesson.classroom
            ]);

            await connection.query(
                'INSERT INTO schedule (teacher_id, group_name, day_of_week, time_slot, course, classroom) VALUES ?',
                [values] // MySQL автоматом обрабатывает массив массивов для VALUES
            );
        }

        await connection.commit();
        return { success: true };
    } catch (error) {
        if (connection) {
            await connection.rollback();
        }
        console.error("Error saving schedule:", error);
        return { success: false, error: "Ошибка сохранения расписания в базе данных." };
    } finally {
        if (connection) {
            connection.release();
        }
    }
}

```

3.4 Сурет. Деректерді алу\өндөу коды

3.2 Академиялық календарь модулы

Бұл модуль оқу жылдарынан маңызды оқығаларын (сессия, мерекелер, аралық бақылау кезеңдері) барлық қолданушыларға көрсетеді, бірақ басқару мүмкіндігін тек Әкімшіге береді.

- Көрсету: Календарь **FullCalendar** JavaScript кітапханасы арқылы іске асырылған. Календарь деректері **auth.js** файлындағы **getAcademicEvents()** функциясы арқылы дерекқордан алғынып, **/api/get_academic_events** маршруты арқылы жіберіледі.(3.5 - 3.6 Сурет)
- Басқару (Тек Әкімші):
 1. Әкімші жаңа оқығаларды қосу, барларын өзгерту және жою мүмкіндігіне ие
 2. Жою процесі **deleteCalendarEvent()** функциясы арқылы жүреді, ол **/api/admin/calendar/delete** (сервердегі **deleteAcademicEvent** функциясына сәйкес) маршрутына **POST** сұранысын жібереді. Бұл, жобаның **рөлдік қауіпсіздік** талаптарын сақтай отырып, маңызды деректерді басқару мүмкіндігін қамтамасыз етеді.(3.7- 3.9 Сурет)

```

async function getAcademicEvents() {
  try {
    const query = `
      SELECT
        id,
        title,
        start_date AS start,
        end_date AS end,
        color,
        description
      FROM
        academic_events
      ORDER BY
        start_date;
    `;

    // ВАЖНО: Мы переименовываем поля в 'start' и 'end',
    // чтобы они сразу соответствовали формату, который ждет FullCalendar!
    const [events] = await pool.query(query);

    // Если end_date установлен в базе как NULL, FullCalendar может быть недоволен.
    // Очистим его здесь, если он не установлен.
    const formattedEvents = events.map(event => ({
      id: event.id,
      title: event.title,
      start: event.start,
      // Если end_date существует, мы его возвращаем, иначе null
      end: event.end ? event.end : null,
      color: event.color,
      description: event.description
    }));
  }

  return { success: true, data: formattedEvents };
}

} catch (error) {
  console.error("DB Error in getAcademicEvents:", error);
  return { success: false, error: "Ошибка БД при загрузке событий календаря." };
}
}

```

3.5 Сурет. Деректерді алу\өндеу

```

else if (req.method === 'GET' && pathname === '/api/calendar/events') {
  const session = req.headers.cookie ? cookie.parse(req.headers.cookie).session : null;
  const user = await checkSession(session);

  // 1. Проверка авторизации: доступен всем авторизованным
  if (!user) {
    res.writeHead(401, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify({ success: false, error: 'Требуется авторизация.' }));
    return;
  }

  try {
    // 2. Вызываем функцию из auth.js
    const result = await getAcademicEvents();

    res.writeHead(200, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify(result));
  }

  } catch (error) {
    console.error("Server error loading academic events:", error);
    res.writeHead(500, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify({ success: false, error: 'Ошибка сервера при загрузке календаря.' }));
  }
  return;
}

```

3.6 Сурет. Сервер сұрауы

```

async function saveAcademicEvent(eventData) {
    const { id, title, start, end, color, description } = eventData;

    try {
        if (id) {
            // Режим обновления существующего события
            const query = `
                UPDATE academic_events
                SET title = ?, start_date = ?, end_date = ?, color = ?, description = ?
                WHERE id = ?;
            `;
            await pool.query(query, [title, start, end || null, color, description, id]);
            return { success: true, message: "Событие успешно обновлено." };
        } else {
            // Режим создания нового события
            const query = `
                INSERT INTO academic_events (title, start_date, end_date, color, description)
                VALUES (?, ?, ?, ?, ?);
            `;
            const [result] = await pool.query(query, [title, start, end || null, color, description]);

            // Возвращаем ID нового события, что важно для клиента
            return { success: true, message: "Событие успешно создано.", newId: result.insertId };
        }
    } catch (error) {
        console.error("DB Error in saveAcademicEvent:", error);
        return { success: false, error: "Ошибка БД при сохранении события." };
    }
}

```

3.7 Сурет. Деректерді сактау коды.

```

async function deleteAcademicEvent(id) {
    try {
        await pool.query("DELETE FROM academic_events WHERE id = ?", [id]);
        return { success: true, message: "Событие удалено." };
    } catch (error) {
        console.error("DB Error in deleteAcademicEvent:", error);
        return { success: false, error: "Ошибка БД при удалении события." };
    }
}

```

3.8 Сурет. Деректерді жою коды

```

else if (req.method === 'POST' && pathname === '/api/admin/calendar/save') {
  const session = req.headers.cookie ? cookie.parse(req.headers.cookie).session : null;
  const user = await checkSession(session);

  // 1. СТРОГАЯ ПРОВЕРКА РОЛИ: ТОЛЬКО АДМИН
  if (!user || user.role !== 'admin') {
    res.writeHead(403, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify({ success: false, error: 'Доступ запрещен. Требуются права Администратора.' }));
    return;
  }

  // 2. Парсинг тела запроса (предполагаем, что у вас есть функция для этого)
  let body = await parseRequestBody(req); // Используйте вашу функцию парсинга

  try {
    const result = await saveAcademicEvent(body);

    res.writeHead(200, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify(result));

  } catch (error) {
    console.error("Server error saving academic event:", error);
    res.writeHead(500, { 'Content-Type': 'application/json' });
    res.end(JSON.stringify({ success: false, error: 'Ошибка сервера при сохранении события.' }));
  }
  return;
}

```

3.9 Сурет. Сервер сұрауы, студент үшін

3.3 Бағалау Және Нәтижелер Модулі

Бұл модуль оқу процесінің нәтижелерін бақылаудың орталық құралы болып табылады

3.3.1 Студенттік Нәтижелерді Көрсету

- Шолу (Summary): Студенттер **renderSummaryTab()** функциясы арқылы курстың жалпы нәтижесін, жинаған баллдарды (**totalEarnedWeight**) және қорытынды буквальық бағаны (**letterGrade**) көре алады.
- Толық Детальдар: **renderDetailsTab()** функциясы жеке бағалау элементтерінің (салмағы, максималды балл, алынған балл) кестесін көрсетеді.
- Іске Асыру Негізі: **getStudentAssessmentDetails** және **getStudentResultsOverview** функциялары дерекқордан алынған деректерді клиенттік жақта өндеп, студентке арналған пайыздық және әріптік бағаны есептейді.(3.10 - 3.12 Сурет)

```

function activateTab(tabId) { ... }

function calculateOverallSummary(assessments) { ... }

function renderWeeklyTable(data) { ... }

function renderStudentResultsTable(assessments, title) { ... }

function renderSummaryTab(summary) {
    return `
        <h2 class="page-header">Общий Результат по Курсу</h2>
        <div class="summary-box">
            <p><strong>Общий Вес Курса:</strong> ${summary.maxWeight}%</p>
            <p><strong>Набранный Вес (Текущий):</strong> ${summary.totalEarnedWeight}%</p>
            <hr>
            <p class="final-grade-line">
                <strong>ИТОГОВЫЙ ПРОЦЕНТ:</strong>
                <span class="final-percent-value">${summary.finalPercent}%</span>
            </p>
            <p class="final-grade-line">
                <strong>БУКВЕННАЯ ОЦЕНКА:</strong>
                <span class="final-letter-value grade-${summary.letterGrade.toLowerCase()}">${summary.letterGrade}</span>
            </p>
        </div>
        <p class="summary-note">
            * Итоговый процент рассчитан на основе проставленных оценок и их весов.
        </p>
    `;
}

```

3.10 Сурет. Нәтижелермен жұмыс жасау функциялары

```

else if (req.method === 'GET' && pathname.startsWith('/api/student/results/details/')) {
    const cookies = cookie.parse(req.headers.cookie || '');
    const session = cookies.session;
    const user = await checkSession(session);

    console.log('[DEBUG] Session ID: ${session}');
    console.log('[DEBUG] User object from checkSession:', user);

    if (!user || user.role !== 'student') {
        res.writeHead(401, { 'Content-Type': 'application/json' });
        res.end(JSON.stringify({ error: 'Доступ запрещен.' }));
        return;
    }

    const parts = pathname.split('/');
    const courseId = parseInt(parts[5]);

    if (isNaN(courseId)) {
        res.writeHead(400, { 'Content-Type': 'application/json' });
        res.end(JSON.stringify({ error: 'Некорректный ID курса.' }));
        return;
    }

    try {
        // user.id берется из активной сессии
        const result = await getStudentAssessmentDetails(user.id, courseId);

        res.writeHead(200, { 'Content-Type': 'application/json' });
        res.end(JSON.stringify(result));
    } catch (error) {
        console.error("Server error fetching student assessment details:", error);
        res.writeHead(500, { 'Content-Type': 'application/json' });
        res.end(JSON.stringify({ success: false, error: "Ошибка сервера при получении деталей оценок." }));
    }
}

```

3.11 Сурет. Сервер сұрауы, студент бағалары

```

async function getStudentAssessmentDetails(studentId, courseId) {
    try {
        const query = ` 
            SELECT
                at.assessment_name,
                at.category,
                at.subcategory,
                at.weight,
                at.max_score,
                sa.score -- Балл студента (может быть NULL, если не проставлен)
            FROM
                assessment_types at
            LEFT JOIN
                student_assessments sa ON at.id = sa.assessment_type_id
                                         AND sa.student_id = ?
            WHERE
                at.course_id = ?
            ORDER BY
                at.category, at.assessment_name;
        `;

        // pool.query должен быть определен в вашем файле auth.js
        const [results] = await pool.query(query, [studentId, courseId]);

        if (results.length === 0) {
            // Возвращаем пустой массив, если нет шаблонов оценок
            return { success: true, data: [] };
        }

        // Группируем данные по категориям для удобства на клиенте
        const groupedData = {};
        results.forEach(item => {
            const category = item.category;
            if (!groupedData[category]) {
                groupedData[category] = [];
            }
            groupedData[category].push(item);
        });

        return { success: true, data: groupedData };
    } catch (error) {
        console.error("DB Error in getStudentAssessmentDetails:", error);
        return { success: false, error: "Ошибка сервера при получении детальных оценок." };
    }
}

```

3.12 Сурет. Деректерді алу\өндөу коды

3.3.2. Оқытушының Баға Қою Функционалы

- Деректерді Жүктеу: Оқытушы курсы және топты таңдағаннан кейін, **/api/get_assessments** маршруты арқылы **getAssessmentTableData** функциясы арқылы бағалау кестесінің құрылымы мен студенттер тізімі жүктеледі.
- Деректерді Сақтау: Оқытушы енгізген бағалар **/api/save_assessments** маршруты арқылы **saveAssessments** функциясына жіберіледі. Бұл функция деректерді мұқият тексеріп, оларды дерекқорға жазады(3.13 - 3.15 Сурет)

```

async function getAssessmentTableData(courseId, groupName) {
  try {
    // 1. Получаем список студентов в нужной группе
    const studentsQuery = ` 
      SELECT
        id AS student_id,
        full_name,
        username
      FROM
        users
      WHERE
        role = 'student' AND group_name = ?
      ORDER BY
        full_name;
    `;
    const [students] = await pool.query(studentsQuery, [groupName]);

    // 2. Получаем все типы оценок для этого курса
    const assessmentsQuery = ` 
      SELECT
        id,
        assessment_name,
        category,
        subcategory
      FROM
        assessment_types
      WHERE
        course_id = ?
      ORDER BY
        category, subcategory, id;
    `;
    const [assessments] = await pool.query(assessmentsQuery, [courseId]);

    // 3. Получаем все существующие оценки для этих студентов
    const gradesQuery = ` 
      SELECT
        student_id,
        assessment_type_id,
        score
      FROM
        student_assessments
      WHERE
        course_id = ? AND student_id IN (?)
    `;
    const studentIds = students.map(s => s.student_id);
    const [gradesRows] = await pool.query(gradesQuery, [courseId, studentIds]);
  }
}

```

3.13 Сурет. Деректерді алу\өндөу

```

function renderTeacherGradingTable(assessments, data, tableId = '') {

  /**
   * Собирает данные из всех полей ввода и отправляет на сервер.
   */
  async function saveGrades(e) {
    e.preventDefault();
    const saveBtn = document.getElementById('saveGradesBtn');
    const saveMessage = document.getElementById('saveMessage');

    saveBtn.disabled = true;
    saveMessage.textContent = 'Сохранение...';
    saveMessage.style.color = '#3498db';

    const courseId = document.getElementById('courseSelect').value;
    const gradesToSave = [];

    // Собираем данные из всех полей ввода на форме
    document.querySelectorAll('.grade-input').forEach(input => {
      const score = parseFloat(input.value);

      // Сохраняем только те оценки, которые введены и находятся в допустимом диапазоне
      if (input.value !== '' && !isNaN(score) && score >= 0 && score <= 100) {
        gradesToSave.push({
          student_id: parseInt(input.dataset.studentId),
          assessment_type_id: parseInt(input.dataset.assessmentId),
          course_id: parseInt(courseId),
          score: score
        });
      }
    });

    try {
      const response = await fetch('/api/teacher/save_assessments', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
        },
        body: JSON.stringify(gradesToSave)
      });
    }
  }
}

```

3.14 Сурет. Деректерді сактау функциясы

```

else if (req.method === 'POST' && pathname === '/api/teacher/save_assessments') {
    const session = req.headers.cookie ? cookie.parse(req.headers.cookie).session : null;
    const user = await checkSession(session);

    if (!user || (user.role !== 'teacher' && user.role !== 'admin')) {
        res.writeHead(403, { 'Content-Type': 'application/json' });
        res.end(JSON.stringify({ error: 'Доступ запрещен.' }));
        return;
    }

    let body = '';
    req.on('data', chunk => {
        body += chunk.toString();
    });

    req.on('end', async () => {
        try {
            const assessments = JSON.parse(body);

            if (!Array.isArray(assessments)) {
                res.writeHead(400, { 'Content-Type': 'application/json' });
                res.end(JSON.stringify({ success: false, error: 'Некорректный формат данных.' }));
                return;
            }

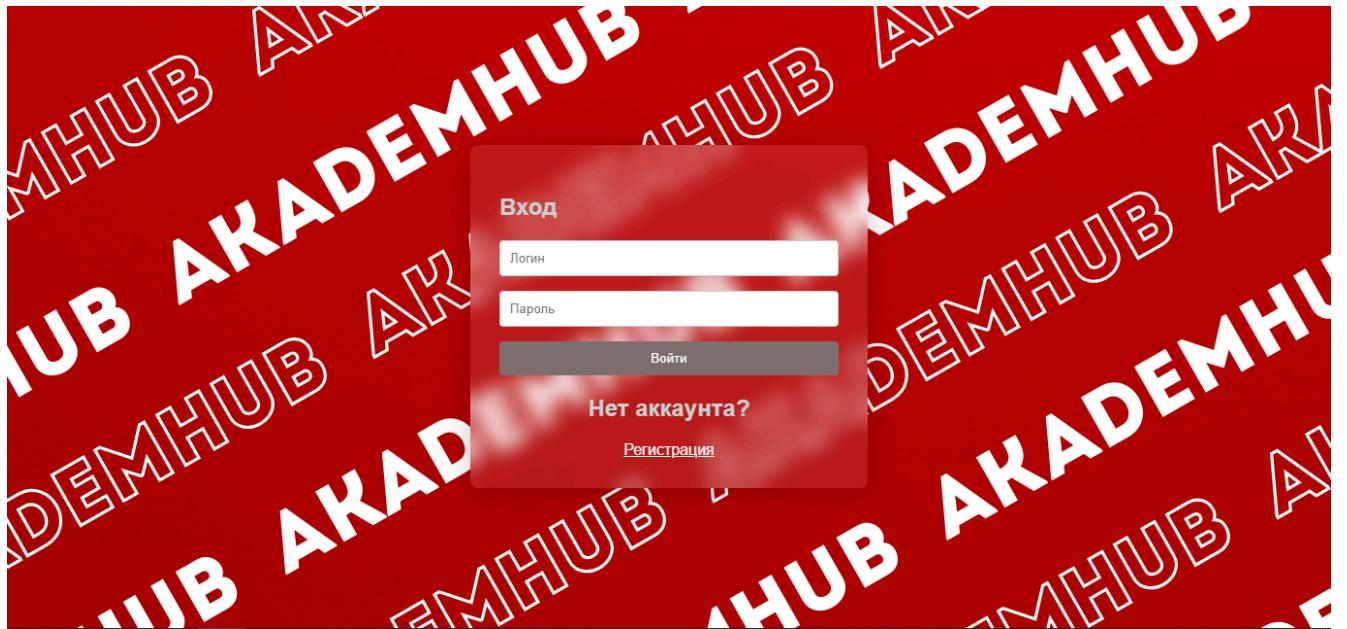
            const result = await saveAssessments(assessments);

            res.writeHead(200, { 'Content-Type': 'application/json' });
            res.end(JSON.stringify(result));
        } catch (error) {
            console.error("Server error processing save assessments:", error);
            res.writeHead(500, { 'Content-Type': 'application/json' });
            res.end(JSON.stringify({ success: false, error: "Ошибка сервера при обработке оценок." }));
        }
    });
}

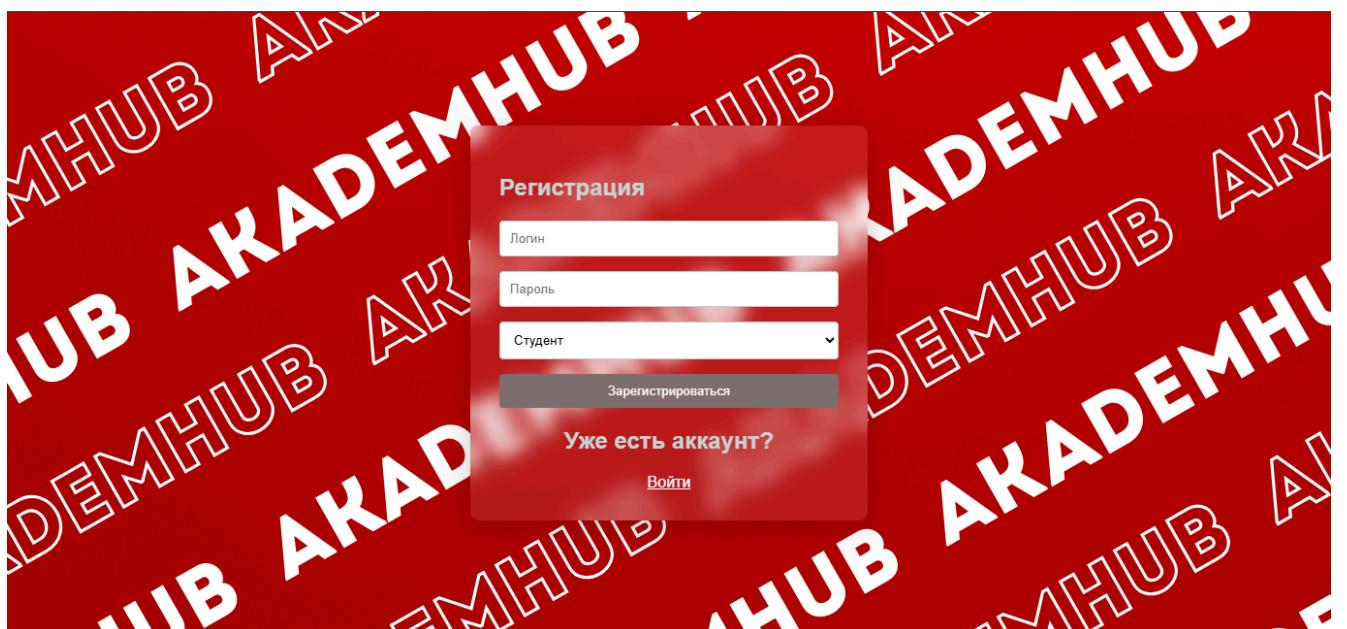
```

3.15 Сурет. Деректерді сактау бойынша сұраныс

Төменде негізгі сайттың толық суреттері көрсетілген. (3.16 - 3.28 Суреттер).



3.16 Кіру бөлімі



3.17 Тіркелу бөлімі



serega
admin

[Главная](#)

[Профиль](#)

[Курсы](#)

[Расписание](#)

[Результаты](#)

[Выйти](#)



Spring 2026 **Erasmus+ Exchange program**
Your next chapter starts abroad!
 Казахстанский Университет науки и технологий приглашает студентов из Европы и мира для участия в международной научно-исследовательской конференции в Астане. Целью конференции является обмен опытом и знаниями между студентами и научными сотрудниками из различных стран. Конференция будет проходить с 28 по 30 октября 2026 года.



Заседание Ученого Совета:
утверждение новых образовательных программ



Студенческий хакатон по искусственному интеллекту собрал рекордное количество

Академический Календарь

<
>
today

| вс | пн | вт | ср | чт | пт | сб |
|----|----|----|----|----|----|----|
| 30 | 1 | 2 | 3 | 4 | 5 | 6 |
| 7 | 8 | 9 | 10 | 11 | 12 | 13 |

00 Неделя 2-Рубежного Контроля

3.18 Басты бет



serega
admin

[Главная](#)

[Профиль](#)

[Курсы](#)

[Расписание](#)

[Результаты](#)

[Выйти](#)

Личный Профиль



serega

[Личные данные](#)
[Сменить пароль](#)
[Сменить фото](#)
[Сообщения](#)

ВХОДЯЩИЕ СООБЩЕНИЯ

Список всех входящих сообщений от пользователей.

| ID | Отправитель | Тема | Дата | Статус | Действия |
|----|-------------|-----------|-----------|--------------|--|
| 11 | Adilzhan | Алертация | 12/4/2025 | НЕ ПРОЧИТАНО | Просмотр Удалить |

3.19 Жек кабинет және Админ үшін хабарламалар

Добавить/Изменить Курс

Название курса: Программирование 1

Описание: Краткое описание

Кредиты: 3

Группа: Выберите группу

Семестр (1-8): 1

ID Преподавателя: HerrBerikuly

Всего курсов: 12

| ID | Курс | Преподаватель ID | Группа | Семестр | Кредиты | Действия |
|----|-------------------------|------------------|--------|---------|---------|--|
| 28 | Английский Интермедиэт | | 33 | 4 | | <input type="button" value="Изменить"/> <input type="button" value="Удалить"/> |
| 26 | Логистика | | 45 | 5 | | <input type="button" value="Изменить"/> <input type="button" value="Удалить"/> |
| 22 | Макро и микро экономика | | 44 | 4 | | <input type="button" value="Изменить"/> <input type="button" value="Удалить"/> |
| 27 | Макро и микро экономика | | 44 | 4 | | <input type="button" value="Изменить"/> <input type="button" value="Удалить"/> |
| 24 | Математика | | 44 | 4 | | <input type="button" value="Изменить"/> <input type="button" value="Удалить"/> |
| 17 | Немецкий В1 | | 33 | 5 | | <input type="button" value="Изменить"/> <input type="button" value="Удалить"/> |
| 20 | Программирование | | 48 | 5 | | <input type="button" value="Изменить"/> <input type="button" value="Удалить"/> |

3.20 Курс қосу бөлімі (Админ)

Создание/Редактирование Расписания

Группа: Логистика День недели: Понедельник

Расписание для Логистика (Понедельник) загружено. Можно редактировать.

| Время | Предмет | Аудитория |
|---------------|-------------|-----------|
| 09:00 - 9:50 | German-D1 | E-105 |
| 10:00 - 10:50 | German-D1 | E-105 |
| 11:00 - 11:50 | | |
| 12:00 - 12:50 | Programming | E-103 |
| 13:00 - 13:50 | Programming | E-103 |
| 14:00 - 14:50 | | |
| 15:00 - 15:50 | | |
| 16:00 - 16:50 | | |

3.21 Сабақ кестесін құру бөлімі(Админ, Оқытушы)

 serega
admin

ВЫСТАВЛЕНИЕ ОЦЕНОК

Курс: Немецкий В1 Группа: Мехатроника Загрузить таблицу

Недельные оценки СРСП Итоговые оценки

Практика (1-15 Неделя)

| Студент | H1 | H2 | H3 | H4 | H5 | H6 | H7 | H8 | H9 | H10 | H11 | H12 |
|----------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----|-----|
| Бакытжанулы Адильжан | 100.0 | 95.00 | 93.00 | 89.00 | 96.00 | 97.00 | 100.0 | 85.00 | 84.00 | 78.00 | | |
| Оспанов Шахмурат | | | | | | | | | | | | |

Лекции (1-15 Неделя)

| Студент | H1 | H2 | H3 | H4 | H5 | H6 | H7 | H8 | H9 | H10 | H11 | H12 |
|----------------------|----|----|----|----|----|----|----|----|----|-----|-----|-----|
| Бакытжанулы Адильжан | | | | | | | | | | | | |
| Оспанов Шахмурат | | | | | | | | | | | | |

3.22.1 Баға қою бөлімі (Админ, Оқытушы)

 serega
admin

ВЫСТАВЛЕНИЕ ОЦЕНОК

Курс: Немецкий В1 Группа: Мехатроника Загрузить таблицу

Недельные оценки СРСП Итоговые оценки

Студент 1-РК 2-РК Сессия

| | | | |
|----------------------|-------|-------|-------|
| Бакытжанулы Адильжан | | | |
| Оспанов Шахмурат | 100.0 | 100.0 | 100.0 |

Сохранить оценки

3.22.2 Баға қою бөлімі



 henBerikuly
 teacher

Преподаваемые курсы

| Название курса | Кредиты | Группы |
|------------------------|---------|-----------|
| Английский Интермедиэт | 4 | Нет групп |
| Немецкий В1 | 5 | Нет групп |

[Главная](#)
[Профиль](#)
[Курсы](#)
[Расписание](#)
[Результаты](#)
[Выйти](#)

3.23 Курстар бөлімі (Оқытушы)



 Berikuly
 student

Личный Профиль


 Berikuly

[Личные данные](#)
[Сменить пароль](#)
[Сменить фото](#)

[Сообщение Админу](#)

СООБЩЕНИЕ АДМИНИСТРАТОРУ

Используйте эту форму для отправки вопросов или сообщений о проблемах администрации.

Тема сообщения:

Опишите подробно вашу проблему...

Текст сообщения:

[Отправить сообщение](#)

[Главная](#)
[Профиль](#)
[Курсы](#)
[Расписание](#)
[Результаты](#)
[Выйти](#)

3.24 Студент үшін Админге хат жазу бөлімі


 Berikuly
student

Ваши Курсы

Выберите Семестр:

| Курс | Преподаватель | Кредиты | Описание |
|-------------------------|---------------|---------|------------------|
| Логистика | Ergoza.A | 5 | Основы логистики |
| Макро и микро экономика | Mukhtarov.D | 4 | Экономика |
| Английский Интермедиэт | herrBerikuly | 4 | Енглиш |

[Главная](#)
[Профиль](#)
[Курсы](#)
[Расписание](#)
[Результаты](#)
[Выйти](#)

3.25 Курстар бөлімі (Студент)


 Berikuly
student

Ваше Расписание (Логистика)

Расписание для группы: Логистика

| Время | Предмет | Аудитория | Преподаватель |
|--------------------|-------------|-----------|---------------|
| Понедельник | | | |
| 09:00 - 09:50 | German-D1 | E-105 | herrBerikuly |
| 10:00 - 10:50 | German-D1 | E-105 | herrBerikuly |
| 12:50 - 13:40 | Programming | E-103 | herrBerikuly |
| 13:50 - 14:40 | Programming | E-103 | herrBerikuly |
| Вторник | | | |
| 12:50 - 13:40 | German-D1 | E-105 | herrBerikuly |
| 13:50 - 14:40 | German-D1 | E-105 | herrBerikuly |
| Среда | | | |
| 12:50 - 13:40 | English | E-106 | herrBerikuly |
| 13:50 - 14:40 | English | E-106 | herrBerikuly |
| Четверг | | | |
| 09:00 - 09:50 | German-D1 | E-105 | herrBerikuly |

[Главная](#)
[Профиль](#)
[Курсы](#)
[Расписание](#)
[Результаты](#)
[Выйти](#)

3.26 Сабақ кесте бөлімі (Студент)

The screenshot shows a student dashboard with a red sidebar on the left containing a user icon and the name "Berikuly student". The main content area is titled "ОБЗОР РЕЗУЛЬТАТОВ ПО КУРСАМ" (Overview of Results by Course). It displays three courses:

- Английский Интермедиэт**: Shows "Нет данных" (No data) and a small green bar chart.
- Логистика**: Shows "Нет данных" (No data) and a small green bar chart.
- Макро и микро экономика**: Shows "Нет данных" (No data) and a small green bar chart.

Below each course, the teacher's name is listed: "Преподаватель: herrBerikuly", "Преподаватель: Ergozha.A", and "Преподаватель: Mukhtarov.D".

Navigation links in the sidebar include: Главная, Профиль, Курсы, Расписание, Результаты, and Выйти.

3.27 Пән бағалары тақтасы (Студент)

The screenshot shows a detailed course result summary for the "Английский Интермедиэт" course. The sidebar is identical to the previous dashboard.

The main title is "ОБЗОР РЕЗУЛЬТАТОВ ПО КУРСАМ" (Overview of Results by Course). A blue button "← Назад к обзору" (Back to Overview) is visible.

Below the title, it says "Детальные оценки по курсу: Английский Интермедиэт".

A navigation bar at the top of the main content area includes tabs: **Общая сводка** (General Summary), Недельные Оценки (Weekly Grades), СРСП (SRSP), and Итоговые Оценки (Final Grades).

The main content area is titled "Общий Результат по Курсу" (Overall Result by Course). It displays the following information:

- Общий Вес Курса: 100.00%
- Набранный Вес (Текущий): 84.30%
- ИТОГОВЫЙ ПРОЦЕНТ: 84.30%**
- БУКВЕННАЯ ОЦЕНКА: B**

A note at the bottom states: * Итоговый процент рассчитан на основе проставленных оценок и их весов.

3.28.1 Толық пән бағалары



Berikuly
student

[← Назад к обзору](#)

Детальные оценки по курсу: Английский Интермедиэт

[Общая сводка](#)

[Недельные Оценки](#)

СРСП

Итоговые Оценки

Главная

Профиль

Курсы

Расписание

Результаты

Выйти

Недельные Оценки (Вес: 1% каждая)

| Неделя | Практика | Лекции |
|----------|-------------|-------------|
| Неделя 1 | 84.00 / 100 | 90.00 / 100 |
| Неделя 2 | 83.00 / 100 | 90.00 / 100 |
| Неделя 3 | 95.00 / 100 | 90.00 / 100 |
| Неделя 4 | 93.00 / 100 | 90.00 / 100 |
| Неделя 5 | 65.00 / 100 | 90.00 / 100 |
| Неделя 6 | 75.00 / 100 | 90.00 / 100 |
| Неделя 7 | 80.00 / 100 | 90.00 / 100 |
| Неделя 8 | 79.00 / 100 | 90.00 / 100 |
| Неделя 9 | 85.00 / 100 | 90.00 / 100 |

3.28.2 Толық пән бағалары

ҚОРЫТЫНДЫ

1. Жобаның Нәтижелері және Мақсаттардың Орындалуы

Осы "Шағын Студенттік Портал" жобасының әзірленуі барысында қойылған барлық негізгі мақсаттар мен міндеттер толығымен орындалды. Жоба Node.js серверлік платформасы, MySQL дерекқоры және Full-stack JavaScript қафидаларына негізделген функционалды, қауіпсіз және тұрақты веб-қосымшаны құруға мүмкіндік берді.

Негізгі нәтижелер:

- Қауіпсіз Рөлдік Модель: Студент, оқытушы және әкімші рөлдері айқын бөлініп, әрбір API маршрутында (мысалы, server.js ішінде) қатаң рұқсат тексеруі жүзеге асырылды. Бұл деректердің рұқсатсыз қолданылуына жол бермейді.
- Тұрақты Сессия Жүйесі: Парольдерді SHA256 арқылы хештеу, сессия ID-лерін кездейсоқ генерациялау (crypto) және Cookie-лерді HttpOnly флагымен пайдалану арқылы жоғары деңгейдегі қауіпсіздік қамтамасыз етілді.
- Деректердің Тұтастығы: SQL транзакцияларының (saveSchedule функциясында) қолданылуы арқасында, кесте деректерін жаңарту кезінде ақпараттың логикалық қайшылыққа ұшырауы толығымен жойылды.
- Модульдік Архитектура: Бекенд логикасының Басқару/Маршруттау (server.js) және DAO/Бизнес-логика (auth.js) деңгейлеріне бөлінуі кодты оқуды, қолдауды және болашақта кеңейтуді жеңілдетеді.

2. Архитектуралық Шешімдердің Тиімділігі

Жобада таңдалған архитектуралық шешімдер оның тиімділігін арттыруды:

- A. Node.js-ті Таңдаудың Тиімділігі

Node.js-ті пайдалану арқылы бір тілдік орта құрылды, бұл әзірлеуді женелдетіп қана қоймай, сонымен қатар сервердің дерекқормен және сұраулармен асинхронды жұмыс істеуі есебінен оның өнімділігін арттыруды. Студенттік портал сияқты I/O операциялары (дерекқорға сұраулар) басым болатын жүйелер үшін бұл ең онтайлы шешім болып табылады.

- Б. MySQL және Пулды Қолдану

MySQL-дің реляциялық тұрақтылығы оқу курсары мен кестелері сияқты құрылымды деректерді сақтауға тамаша сәйкес келді. Қосылым пулын қолдану сервердің тұрақтылығын және көптеген пайдаланушы бір мезгілде кірген кездегі реакция жылдамдығын қамтамасыз етті.

3. Келешектегі Даму Жоспарлары

Бұл жоба жұмыс істейтін негізгі үлгі болып табылады. Оны одан әрі дамыту үшін келесі бағыттар ұсынылады:

- Қауіпсіздікті Арттыру: Парольдерді хештеу үшін SHA256 орнына bcrypt немесе scrypt сияқты заманауи және баяу алгоритмдерге көшу.
- Функционалды Кеңейту: Бағалау жүйесін (бағалар, рейтингтер) немесе хабарландыру жүйесін (мысалы, WebSockets арқылы) қосу.
- Frontend-ті Жаңарту: Клиенттік JS-тің орнына React, Vue немесе Angular сияқты заманауи JavaScript фреймворктарін пайдалану арқылы интерфейстің күрделілігін және жылдамдығын арттыру.

Қорытынтылай келе, "Шағын Студенттік Портал" жобасы Node.js, MySQL және қауіпсіз кодтау әдістерін тиімді қолданудың сәтті мысалы болып табылады. Жоба қойылған білім беру және техникалық міндеттерді толықтай шешті.

ПАЙДАЛЫНЫЛҒАН ӘДЕБИЕТТЕР

1. Бұркітбаев, М. Ә. Ақпараттық жүйелердің негіздері. — Алматы: Эверо, 2020.
2. Ермекбаев, Б. С. Веб-технологиялар және веб-бағдарламалар негіздері. — Алматы: Қазақ университеті, 2021.
3. W3Schools. “HTML, CSS және JavaScript Tutorials.” — <https://www.w3schools.com>
4. Mozilla Developer Network (MDN). “JavaScript Documentation.” — <https://developer.mozilla.org>
5. FreeCodeCamp. “Responsive Web Design and Front-End Development.” — <https://www.freecodecamp.org>
6. Khan Academy. “Intro to JS: Drawing & Animation.” — <https://www.khanacademy.org/computing>
7. ISO/IEC 25010:2011 — Systems and Software Engineering – System and Software Quality Models.
8. Stack Overflow Community. “Frontend Development Discussions and Code Examples.” — <https://stackoverflow.com>