ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
коммерциялық емес акционерлік қоғамы
«ҒҰМАРБЕК ДАУКЕЕВ АТЫНДАҒЫ АЛМАТЫ ЭНЕРГЕТИКА ЖӘНЕ БАЙЛАНЫС УНИВЕРСИТЕТІ»
IT-инжиниринг кафедрасы

**ҚОРҒАУҒА ЖІБЕРІЛДІ**
Кафедра меңгерушісі
PhD, доцент А.А. Досжнова
_____ «___» _____ 2020 ж.

# ДИПЛОМДЫҚ ЖОБА

Тақырыбы: Кәсіпорын қызметкерлерінің жұмыс уақытын бақылаудың ақпараттық жүйесін құру .
Мамандығы: 5В070300 – «Ақпараттық жүйелер»
Орындаған: Карин А.Ш. Тобы: ИСк-16-1
Ғылыми жетекші: PhD, доцент Кожамкулова Ж.Ж.

Кеңесшілер:
Экономикалық бөлім:
э.ғ.к., асс. профессор Габелашвили К.Р. _____ «___»_____2020 ж.
(ғылыми дәрежесі, атағы, Т.А.Ж) (қолы)

Өміртіршілік қауіпсіздігі:
аға оқытушы Бегимбетова А.С. _____ «___»_____2020 ж.
(ғылыми дәрежесі, атағы, Т.А.Ж) (қолы)

Есептеу техникасын қолдану:
аға оқытушы Айткулов Ж.С. _____ «___»_____2020 ж.
(ғылыми дәрежесі, атағы, Т.А.Ж) (қолы)

Норма бақылаушы:
аға оқытушы Абсатарова Б.Р. _____ «___»_____2020 ж
(ғылыми дәрежесі, атағы, Т.А.Ж) (қолы)

Сын-пікір беруші:
_____ _____ «___»_____2020 ж.
(ғылыми дәрежесі, атағы, Т.А.Ж) (қолы)

Алматы 2020

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ
коммерциялық емес акционерлік қоғамы
«ҒҰМАРБЕК ДАУКЕЕВ АТЫНДАҒЫ АЛМАТЫ ЭНЕРГЕТИКА ЖӘНЕ БАЙЛАНЫС УНИВЕРСИТЕТІ»

Басқару жүйелері және ақпараттық технологиялар институты

IT-инжиниринг кафедрасы

Мамандығы 5В070300 – «Ақпараттық жүйелер»

Дипломдық жобаны орындауға берілген
**ТАПСЫРМА**

Студент: Карин Арман Шоканулы

Жоба тақырыбы: Кәсіпорын қызметкерлерінің жұмыс уақытын бақылаудың ақпараттық жүйесін құру.

«__» _____ 20____ ж. №_____ университет бұйрығы бойынша бекітілген.

Аяқталған жұмысты тапсыру мерзімі: «_____» _____2020 ж.

Жобаға бастапқы деректер (талап етілетін жоба нәтежелерінің параметрлері және нысанның бастапқы деректері): Бұл дипломдық жоба C# тілін пайдаланып, жұмысшылардың жұмыс уақытын бақылауақпараттық жүйені құру.

Диплом жобасындағы әзірленуі тиіс сұрақтар тізімі немесе диплом жобасының қысқаша мазмұны:

а) Пәндік саланы талдау;
б) Жобалау бөлімі;
в) Программалық қамтаманы құру;
г) Экономикалық бөлім;
д) Өміртіршілік қауіпсіздігі;
е) А қосымшасы. Программа мәтіні;

Графикалық материалдар тізімі (міндетті сызбалар дәл көрсетілуі тиіс):
47 сурет, 12 кесте.

Негізгі ұсынылатын әдебиеттер:

1 Маркин, А.В. PHP web-бағдарламалау негіздері: учеб. пособие / А.В. Маркин. – М.: Диалог-МИФИ, 2012 жыл.

2 Когтзолл Д. PHP5. Толық жетекшілік. – М.: Вильямс, 2010.

3 CSS туралы оқулық – Электронды оқулық: http://www.wisdomweb.ru/CSS.

4 HTML туралы оқулық – Электронды оқулық: https://html5book.ru/

Жоба бойынша бөлімшелерге қатысты белгіленетін кеңесшілер

| Бөлімшелер | Кеңесшілер | Мерзімі | Қолы |
|---|---|---|---|
| Экономикалық бөлім | Габелашвили К.Р. | 15.04.2020 – 30.04.2020 | |
| Өміртіршілігі қауіпсіздігі | Мусаева Ж.К. | 15.04.2020 – 30.04.2020 | |
| Бағдарламалық қамтама | Айткулов Ж.С. | 13.05.2020 – 18.05.2020 | |
| Норма бақылау | Абсатарова Б.Р. | 13.05.2020 – 18.05.2020 | |

Диплом жобасын дайындау
КЕСТЕСІ

| № р/с | Тарау аттары, әзірленетін сұрақтардың тізімі | Жетекшіге ұсыну мерзімдері | Ескерту |
|---|---|---|---|
| 1 | Теориялық бөлім | 17.02.2020 - 16.03.2020 | |
| 2 | Бағдарламалық қосымшаны жобалау | 17.03.2020 - 05.04.2020 | |
| 3 | Қосымша әзірлеу бөлімі | 06.04.2020 - 09.05.2020 | |

Тапсырманың берілген уақыты «___» _____ 20____ж.

Кафедра меңгерушісі _____ Досжанова А.А.

Жоба жетекшісі _____ Кожамкулова Ж.Ж.

Орындалатын тапсырманы
қабылдаған студент _____Карин А.Ш.

## АҢДАТПА

Бұл дипломдық жобада өз қызметкерлеріне сағат сайын төлейтін компаниялар үшін артық шығындар мәселесі қарастырылады. Көбінесе мұндай компаниялардың жұмысшылары жұмыс уақытының есебінен алдайды, жұмыс уақытына кешігіп келеді және оны көбінесе ертерек қалдырады, ал менің жұмысым мәселенің шешімін табады. Бұл бағдарламалық қамтамасыз ету қызметкерлердің келу және кету уақытын ескереді, осылайша компанияның жақсы қызметкерлерді таңдап, олардың сұрыпталуына көмектесетін қажетсіз шығындардың деңгейін төмендетеді.

Жобадағы бағдарлама c # тілінде Visual студиясының көмекші бағдарламасында жазылды және дерекқорды құре және басқаре үшін, бағдарламаны жергілікті MAMP хостын құру үшін қолданылды, содан кейін phpMyAdmin дерекқорды конфигурациялады.



## АННОТАЦИЯ

В данном дипломном проекте рассматривается вопрос о лишних затратах для компаний, которые оплачивают своим сотрудникам каждый час. Зачастую работники таких компаний обманывают за счет рабочего времени, опаздывают на рабочее время и зачастую оставляют его раньше, а моя работа решит проблему. Это программное обеспечение учитывает время прибытия и убытия сотрудников, таким образом, снижает уровень ненужных затрат, которые помогут компании выбрать лучших сотрудников и получить их сортировку.

Программа в проекте была записана в вспомогательной программе студии Visual на языке c # и использована для создания локального хостера MAMP для создания и управления базами данных, а затем конфигурирована база данных phpMyAdmin.



## ANNOTATION

This diploma project addresses the issue of extra costs for companies that pay their employees every hour. Often employees of such companies cheat at the expense of working time, are late for work time and often leave it earlier, and my work will solve the problem. This software takes into account the arrival and departure times of employees, thus reducing the level of unnecessary costs that will help the company select the best employees and get them sorted.

The program in the project was written in the visual Studio helper program in c # and used to create a local MAMP host for creating and managing databases, and then configured the phpMyAdmin database.

**Мазмұны**

# Кіріспе

Сағаттық төлем - бұл уақытқа негізделген төлем формасының ерекше жағдайы. Ол қызметкердің жұмысын қалыпқа келтіру қиын болған жағдайда қолданылады.

Бір сағат ішінде жұмысшы жасаған бөліктердің санын есептей алу мүмкін, бірақ мұғалімнің, заңгердің жұмысын қалай бағалауға болады? Сондықтан қызметкердің біліктілігі де ескеріледі.

Сағаттық жалақы жұмыс беруші үшін қаншалықты пайдалы?

Біріншіден, жұмыс күнінен айырмашылығы, жұмыс уақыты әрдайым бірдей болады. Бұл жағдайда жұмысшының жұмыс уақытының құнын белгілеп, әртүрлі себептермен жұмыс орнында болмаған уақытты шегеріп, қанша еңбекақы тапқанын нақты есептей алу қажет.

Екіншіден, сағаттық жалақы стандартты емес кесте (толық емес жұмыс күні немесе апта, толық емес жұмыс уақыты немесе икемді сағаттар) бар жұмысшыларға еңбекақы төлеу үшін оңтайлы болып табылады.

Үшіншіден, жұмыс беруші ақша үнемдейді. Егер ол толығымен бос болмаса және жай кеңседе отырса, қымбат маманның уақытын төлеу тиімсіз. Сағаттық жалақымен компаниямен қызметкерлерге нақты жұмыс істеген уақытына ғана төленеді.

Төртіншіден, сағаттық төлем жүйесі қарапайым және жылдам есептеледі. Діни жұмыс айтарлықтай төмендейді, қақтығыс қаупі азаяды.

Дипломдық жұмысымның мақсаты - қызметкерлердің уақыттық есебін жасау және енгізу. Осы жүйені қолданудың арқасында компания автоматтандырылған операциялардың күрделілігін, атап айтқанда қызметкерлердің уақытын есепке алуды азайту аясында экономикалық пайда алады.

Бағдарламаның бұл түрін құру идеясы тәжірибелік сабақтар өтілген компания негізінде пайда болды, менің байқауларым бойынша көптеген қызметкерлер таңертең де, түскі үзілістен кейін де кешіккен. Бұл бағдарламаның көмегімен қызметкердің жұмыс орнына қашан және қай уақытта кеткенін көруге болады, осылайша жұмыс берушінің шығындары қажетсіз шығындардан қысқарады.

# 1 Кәсіпорындағы жұмысшылардың жұмыс атқаруының бақылау тәсілдері

## 1.1 Дипломдық жоба тақырыбының өзектілігі

Кейде жұмысшыға көп жұмыс істеуге тура келеді, бірақ жұмыс күнінің соңында ештеңе дайын емес - біз жұмыс уақытының не екені айтуға қиын және жұмысшы өз қателіктерін сабаққа алмайды.

Басқару жүйелері - бұл нақты уақытты жұмсай отырып, кәсіби өсуге зиян келтіретін өндірістік емес әдеттерді өзгертетін нәрсе.

Табысты адамдар бұдан былай жұмыс жасамайды, оларда уақытты басқару жағынан ешқандай қиындықтар болмайды. Қызметкерлерді бақылау жүйелері өнімділікті арттыру және өзін-өзі жетілдіру үшін барлық қажетті ақпаратты ұсынады. Бұл функция қандай міндеттерді шешті?

Интернеттегі компьютерлік бақылау функциясы бақылаудан басқа, жаңа қызметкерлерді оқытуға және үйлестіру әрекеттерін жасауға көмектеседі.

Жоғары жүйелі, өршіл кәсіпқойлар командасы басқару жүйелерінің басты артықшылықтарының бірі болып табылады.

Жақсы қызметкерлерді анықтауға көмектеседі. Егер адам басқаларға қарағанда көп жұмыс жасаса және оның үлесі елеусіз қалса, бұл ынта-жігерді айтарлықтай төмендетеді. Ақыр соңында, біреу барған сайын нәтижелі жұмыс істейді және мұны атап өткен жөн. Мониторинг қызметтерінің мәліметтеріне сүйене отырып, бастығы негізгі үлес қосатын ең өнімді қызметкерлерді анықтай алады. Сіздер байқағандай, кейбіреулері бүкіл жұмыс күндерін коммуналдық қызметпен өткізеді, ал басқалары жиі және ұзақ үзілістер жасайды, әлеуметтік желілерде, интернет-дүкендерде және т.б. Ең жақсысын беруге тырысатын және оларды бағалайтын команда мүшелерін, сондай-ақ жеңдерімен жұмыс жасайтындарды елемеу әділетсіз болады.

Қызметкерлерді уақытты бақылау жүйесімен таныстыра отырып, мұндай құрал қызметкердің немесе жұмыс берушінің пайдасына кететін қателіктерді жойып, жалақы төлеу кезінде қызметкерлердің жұмыс уақытын автоматты түрде есептейтінін түсіндіруге тырысыңыз. Бағдарлама қосымша екі сағатты санауға азғырады және барлық өңдеуді нақты бекітеді. Жұмыс уақыты туралы есеп қызметкерлердің жұмысын едәуір жеңілдетеді: оларды қолмен жасаудың қажеті жоқ, менеджер өз кезегінде бәрін тексеруге жеңілірек болады. Сонымен қатар, сіз почтаға Excel форматында есептерді жіберуді теңшей аласыз. Алынған мәліметтер жалақыны көтеруді талқылау кезінде де ескерілуі мүмкін; бұл ең жақсы мамандарға олардың күш-жігері босқа кетпейтіндігінің кепілі.

Менеджерлерге қанша уақыт қажет болатындығын түсінуге көмектеседі. Тәжірибелі менеджер мұны естігенде, олар «бұл кішігірім тапсырмаға бес минут уақыт кетеді» дейді, ол бірден дабыл қағады. Шынында да, кішкене жақсартуға көбінесе бірнеше сағат кетеді.

Себебі, көбіне жұмыс процесін ұйымдастырып, тапсырмаларды үлестіруге болатын менеджерлер белгілі бір тапсырмаларға қанша уақыт жұмсауға болатыны туралы шешім шығармайды. Нәтижесінде жұмысшылар көбінесе нақты емес уақыт аралығында жұмыс істеуге мәжбүр болады.

Қызметкерлердің мониторингі, басқалармен қатар, басшылық жұмыстың ерекшелігіне тереңірек үңілу керек деп болжайды. Командаңызға тапсырманы орындауға кететін уақытты қадағаласаңыз, жұмыс көлемі туралы жақсырақ түсінік алуға көмектесетінін түсіндіріңіз.

Нәтижесінде бұдан әрі шынайы емес үміттер мен мүмкін емес мерзімдер болмайды.

Команда мүшелерін уақытты бақылау жүйесімен таныстыра отырып, қандай-да бір оқиғадан бастаңыз, олар өздері байланыстыра алады. Содан кейін қадағалау жүйелері болашақта жағымсыз оқиғалардың алдын алуға қалай көмектесетінін түсіндіріңіз. Қоқыстардан үнемдейді

Менеджерлер өз қызметкерлерінен жақсы демалады деп үміттенеді - демек, көбірек жұмыс қанағаттануы мен жоғары өнімділік. Алайда, ірі және орта компаниялардың проблемасы басшылыққа әр қызметкердің жұмысын тексеру және олардың өздеріне қамқорлық жасау, уақтылы үзілістер жасау және жұмысты уақтылы тоқтату болып табылады. Егер жұмысшы күні бойы 16 сағат жұмыс жасайтын болса, бұл денсаулық үшін ғана емес, келесі күні, өте ұзақ ауысымнан кейін, нәтижелігіне де әсер етеді.

Егер сіз кеңседе ұзақ уақыт тұрсаңыз, бұл көп нәрсені істеуге көмектеседі, бірақ сіз шаршап көрінетіндіктен, келесі күні шоғырлану өте қиын болады.

Командаңызға уақытты бақылау - бұл команданың әл-ауқатын жақсартуға арналған басқа басқару және кадрлық құрал екенін түсіндіріңіз. Кеңседе ұзақ болуға ешкім тыйым салмайды, бірақ күн сайын екі ауысым ешкімге пайда әкелмейді.

Өнімділіктің толық көрінісін көруге көмектеседі

Қызметкерлер қадағалау жүйелеріне мән бермейді, ал микро басқаруды болса ескереді. Сіз таңдаған бағдарламалық жасақтаманың жалпыланған мәліметтермен қамтамасыз етуі маңызды және сізге егжей-тегжейлерді ашпау үшін толық суретті көруге мүмкіндік береді.

### 1.2 Әзірленген бағдарламалық жасақтаманың мақсаты

Жұмыс уақытының есебі кәсіпорында, бір немесе өзге де нысанда бар бұрыннан. Әдетте, бұл өткізу жүйесі немесе қағаз табельдер. Алайда, бұл әдістеме қызметкердің жұмыста болу фактісін ғана белгілеуге мүмкіндік береді. Электрондық есеп бағдарламасының кеңейтілген функционалы бар. Ол күн мен сағатты енгізу үшін аймақтармен кестені білдіреді. Осы деректерді бекіткеннен кейін, сіз жүйелік талдау құралын аласыз. Есепті кезеңдегі кестелерді салыстыра отырып, нәтижеліліктің жалпы көрінісін алуға болады. Мысал үшін біз жұмысшыны аламыз. Ол келіп жұмысқа кешігу және

кетіп әрқашан уақытында. Бұл стандартты табель. Бірақ, ол күн ішінде не істеп түсіну маңызды. Тұрақты перекуралар, әлеуметтік желілерде отыру және әріптестермен қарым – қатынас-компаниялар табыс әкелмейді. Сондықтан, маңызды емес фактісі болған қызметкерінің Х ал оның өнімділігі. Егер персоналдың жұмыс уақытын есепке алу бағдарламалық қамтамасыз ету арқылы жүргізілсе, сіз қандай жобалармен жұмыс істейтінін көре аласыз. Сонымен қатар, ол қандай міндеттерді орындайды, жұмыстағы үзіліс ұзақтығы мен интервалы, қандай да бір тапсырманы орындауға қажет болған уақыт. Жұмыс күнінің соңында х қызметкерінің өзі мен оның жұмыс берушісі картинаның толық болуын көріп, кеңседе өткізілген уақыттың тиімділігін бағалай алады. Жұмыс уақытының мұндай есебі кәсіпорында үдерістерді айтарлықтай оңтайландыруға мүмкіндік береді. Сонымен қатар, БОЙЫНША енгізуге мүмкіндік береді жүйесіне жіберу туралы деректер, күндер, отгулах және т. б. Байланысты жылдам көрсету жүйесінде, осы ақпаратты қолдан жасау мүмкін емес немесе түзету. Осылайша, қызметкерлері ұйымдастыра алмайды, өзіне бірнеше ақылы демалыс фирманың есебінен. Жұмыс уақытын есепке алу бағдарламасы автоматты түрде пайдаланылған демалыс күндері мен оның қалдығының есебін жүргізеді. Бөлінуін қадағалайды жұмыс, тіркейді уақыты. Мұндай мониторинг орындалған міндеттердің көрсеткішімен сәйкес келеді. Бұл өңдеу жұмыс үдерісімен объективті негізделді ма, мысалы, шұғыл міндеттерді орындау немесе Х қызметкері жұмыс уақытын тиімсіз пайдаланды ма, сондықтан үлгермейді. Осыны негізге ала отырып, шешім қабылдайды көтермелеу туралы қызметкердің немесе қолдануға санкция. БҚ көмегімен жұмыс уақытын есепке алу өз қызметкерлерінің өнімділігіне жүйелі талдау жүргізуге және проблемалық салаларды атап өтуге мүмкіндік береді. Есепті кезеңдегі кестелерді салыстыру барлық бағыттар бойынша жалпы көрініс көруге, жобалардың маржиналығын және бюджетті бағалауға мүмкіндік береді.

### 1.3 Іске асыру ортасының компоненттерін таңдау

Осы дипломдық жобаны жазу кезінде келесі компоненттер қолданылды:

1) Microsoft Visual studio бағдарламалық жасақтама құралы көмегімен интерфейсі құрылды.

2) Apache интернет сервері. Apache-бүгінгі таңда ең танымал сервер. Өйткені ол басқару оңай деп саналады және ол тегін таратылады.

3) PHP интернет тілі. PHP тілі бар кезінде өзін ыңғайлы, салыстырмалы жылдам және үлкен мүмкіндік береді.

4) phpMyAdmin PHP-де жазылған және MySQL ДҚБЖ-ны басқаруға арналған веб-интерфейсті ұсынатын ашық бастапқы бағдарлама.

5) MySQL деректер базасы. MySQL-бұл PHP-мен бірге жиі қолданылатын деректер қорын басқарудың танымал жүйесі (ДББЖ). Му SQL кестелерді жасау және көру үшін өз нысандары жоқ.

## 1.4 Іске асыруға арналған технологиялар мен құралдар

### 1.4.1 PHP бағдарламалау тілі

PHP атауы - "PHP: Hypertext Preprocessor"дегенді білдіретін рекурсивті аббревиатура. Бастапқыда веб-беттерді әзірлеуді жеңілдету үшін Perl үстінде қондырма ретінде құрылды.

PHP-веб-серверде HTML беттерін генерациялау және деректер базасымен жұмыс істеу үшін құрылған бағдарламалау скрипті тілі. Қазір хостингтің басым көпшілігіне қолдау көрсетеді. Веб – сайттарды жасау үшін LAMP-"стандартты" жиынтығына кіреді.

PHP желісіне арналған бағдарламалау саласында-өзінің қарапайымдылығының, орындау жылдамдығының, бай функционалдылығының және PHP лицензиясы негізінде бастапқы кодтардың таралуының арқасында танымал скрипті тілдердің бірі (JSP және ASP-де пайдаланылатын тілдермен қатар). PHP ядроның және қосылатын модульдердің, "кеңейтулердің" болуымен ерекшеленеді: деректер базасымен, сокеттермен, динамикалық графикамен, криптографиялық кітапханалармен, PDF форматындағы құжаттармен және т. б. жұмыс істеу үшін. Жүздеген кеңейтулер бар,бірақ стандартты жеткізуге өзін жақсы көрсеткен бірнеше ондаған ғана кіреді. PHP интерпретаторы веб-серверге немесе осы сервер үшін арнайы жасалған модуль арқылы (мысалы, Apache немесе IIS үшін) немесе CGI-қосымша ретінде қосылады. Сонымен қатар, ол UNIX, Linux, Windows және Mac OS X операциялық жүйелерінде әкімшілік тапсырмаларды шешу үшін пайдаланылуы мүмкін.

Тілдің синтаксисі Си тіліне ұқсас. Кейбір элементтер сияқты ассоциативті массивтер және foreach циклы Perl алынған.

PHP - дің ең күшті және маңызды мүмкіндігі-деректер қорымен интеграциялау деңгейі. Деректер базасымен жұмыс істейтін веб-бетті жазу өте оңай. Қазіргі уақытта келесі деректер қоры қолдау табады: Oracle, Adabas D, Sybase, FilePromSQL, Velocis, MySQL, Informix, Solid, dBase, ODBC, Unix dbm, PostgreSQL.

Мысалы, JavaScript, PHP сценарийлері серверде орындалады. Серверді HTML файлдары PHP процессорымен өңделетін етіп теңшеуге болады, сондықтан клиенттер тіпті олардың әдеттегі HTML файлын немесе скриптті орындау нәтижесін ала алмайтынын біле алмайды.

PHP игеру үшін өте оңай, бірақ кәсіби бағдарламашылар сұраныстарын қанағаттандыра алады.

PHP, негізінен, web-серверлер ортасында жұмыс істеуге арналған болса да, оның қолдану аумағы тек қана мұнымен шектелмейді.

PHP қолдану аймағы сервер жағында жұмыс істейтін скрипттерді жазуға бағытталған; осылайша, PHP кез келген басқа CGI бағдарламасын орындауға қабілетті, мысалы, пішін деректерін өңдеу, динамикалық беттерді жасау немесе жіберу және cookies қабылдау. Бірақ PHP басқа да көптеген тапсырмаларды орындай алады.

PHP пайдаланатын үш негізгі аймақ бар:

- сервер жағында орындау үшін скрипттерді жасау. PHP ең кең қолданылады дәл осылай. Қажет болуы мүмкін барлық PHP парсері (CGI бағдарламасы немесе серверлік модуль түрінде), веб-сервер және браузер. Браузерде PHP скрипттерін орындау нәтижелерін көру үшін жұмыс істейтін веб-сервер және PHP орнатылған қажет;

- командалық жолда орындау үшін скрипттер жасау. Веб-серверге және браузерге қарамастан іске қосылатын PHP сценарийін жасауға болады. Барлық қажет-парсер PHP. PHP қолдануының мұндай тәсілі Windows платформаларында cron (*nix немесе Linux платформаларында) немесе тапсырмаларды жоспарлаушы (Task Scheduler) көмегімен жүйелі түрде орындалуы тиіс скрипттер үшін өте қолайлы. Бұл скрипттер қарапайым мәтіндерді өңдеу міндеттеріне пайдаланылуы мүмкін;

- клиенттің жағында орындалатын GUI қосымшаларын құру.

Сонымен қатар, PHP мұндай қосымшаларды жасау үшін ең жақсы тіл емес, бірақ егер программист PHP өте жақсы біледі және оның кейбір мүмкіндіктерін өзінің клиент-қосымшаларында қолданғысы келсе, осындай қосымшаларды жасау үшін PHP-GTK пайдалану мүмкіндігі бар. Осындай жолмен кросс-платформалық қосымшаларды жасауға болады. PHP-GTK PHP кеңейтімі болып табылады және PHP дистрибутивімен бірге жеткізіледі.

PHP көптеген операциялық жүйелер үшін қол жетімді, оның ішінде Linux, Unix көптеген модификациялары (HP-UX, Solaris және OpenBSD сияқты), Microsoft Windows, Mac OS X, RISC OS және т.б. Сонымен қатар, PHP-те Apache, Microsoft Internet Information Server, Personal Web Server, Netscape және iPlanet серверлері, Oreilly Website Pro, Caudium, Xitami, Omnihttpt және т.б. сияқты көптеген заманауи веб-серверлердің қолдауы бар. Көптеген PHP серверлері CGI стандартын қолдайтын басқа да модуль ретінде жеткізіледі, PHP CGI процессоры ретінде жұмыс істей алады.

Осылайша, PHP бағдарламалау тілін таңдай отырып, программист Операциялық жүйе мен веб-серверді таңдау еркіндігін алады.

### 1.4.2 Веб-сервер Apache

Apache-бұл клиенттерден, әдетте веб-браузерлерден HTTP-сұрауларды қабылдайтын және оларға HTTP-жауаптарды беретін сервер, әдетте HTML-парағымен, бейнесімен, файлымен, медиа-ағынымен немесе басқа да деректермен бірге.

Әдетте веб-браузерлер болып табылатын Клиент веб-серверге URL-адрестерімен белгіленген ресурстарды алуға сұрау жібереді. Ресурстар-Бұл

HTML-беттер, суреттер, файлдар, медиа ағындары немесе клиентке қажетті басқа да деректер. Жауап ретінде веб-сервер клиентке сұралған деректерді береді. Бұл алмасу HTTP хаттамасы бойынша жүреді.

Apache веб-серверінің негізгі артықшылықтары-конфигурацияның сенімділігі мен икемділігі. Бұл веб-сервер деректерді ұсыну үшін сыртқы модульдерді қосуға, пайдаланушыны аутентификациялау үшін ДҚБЖ пайдалануға, қателер және т. б. туралы хабарламаларды түрлендіруге мүмкіндік береді.

Apache мәтіндік конфигурациялық файлдар арқылы теңшейді. Негізгі параметрлер "әдепкі" деп орнатылған және көп жағдайда жұмыс істейтін болады. Егер штаттық "Апачтың" функционалдығы жеткіліксіз болса, онда Apache Group және бөгде әзірлеушілер жазған әр түрлі модульдерді пайдалануға болады. Маңызды артықшылығы болып табылады жасаушылар белсенді түрде қатынасып және пайдаланушылармен әрекет жасайды, барлық қате туралы хабарлар.

Apache орындауға болатын ең қарапайым функция – серверде тұрып, әдеттегі HTML-Сайтқа қызмет көрсету. Белгілі бір бетке сұраныс алған кезде сервер оған жауап ретінде браузерге жібереді. Сұраныс ретінде браузердің адрестік жолында терілген мекен-жай шығады.

Дизайн және сайттың функционалдық бөлігін бөлу үшін, сондай-ақ статикалық нысандардың өзгеруін жеңілдету үшін SSI технологиясы бар. Ол қайталанатын барлық ақпаратты бір файлға (мысалы, top.inc), содан кейін бетке оған сілтеме қою. Содан кейін, осы ақпаратта бірнеше жолды өзгерту қажет болса, оларды тек бір файлда ғана өзгерту керек. Apache сервері бұл технологияны қолдайды және серверлік қосылыстарды толық көлемде пайдалануға мүмкіндік береді.

Web-сервер функцияларын компьютер өзі емес, онда орнатылған бағдарлама, яғни пайдаланушының браузері web-серверге қосылған кезде және GET тақырыбын жібереді (бұл файлды жіберу өтініші), оның сұрауын дәл Apache өңдейді. Apache get атауында көрсетілген файл бар-жоғын тексереді және Бар болса, оны браузер тақырыптарымен бірге жібереді.

Apache сервері виртуалды серверлерді (хостарды) қолдай бастаған алғашқы серверлердің бірі болды. Бұл мүмкіндік Бір физикалық серверде бірнеше толыққанды сайттарды орналастыруға мүмкіндік береді. Олардың әрқайсысының өз домені, әкімшісі, IP мекенжайы және т.б. болуы мүмкін.

Apache-да CGI және PHP технологиясын қолдау, сондай-ақ басқа тілдерді қосу мүмкіндігі бар. Бұл динамикалық интернет-беттермен жұмыс істеуді айтарлықтай жеңілдетеді (олар шын мәнінде, бүгінгі күні барлық веб-беттер болып табылады).

### 1.4.3 MySQL ДБ басқару жүйесі

MySQL – бұл PHP-мен бірге жиі қолданылатын деректер қорын басқарудың танымал жүйесі (ДББЖ).

Деректер базасы құрылымдық деректер жиынтығы болып табылады. Бұл деректер кез – келген болуы мүмкін-алдағы сатып алудың қарапайым тізімінен сурет галереясының экспонаттарының тізіміне немесе корпоративтік желідегі ақпараттың үлкен санына дейін. Компьютерлік деректер базасында сақталатын деректерді жазу, таңдау және өңдеу үшін MySQL бойынша сияқты деректер базасын басқару жүйесі қажет. Деректер қорын басқару есептеулерде орталық рөл атқарады. Мұндай басқару әртүрлі болуы мүмкін-жеке утилиталар түрінде де, басқа қосымшалардың құрамына кіретін код түрінде де.

MySQL – бұл реляциялық деректер қорын басқару жүйесі. Бұл деректер базасындағы деректер барлық жинақта емес, жеке кестелерде сақталады, соның арқасында жылдамдық пен икемділік ұтысына қол жеткізіледі. Кестелер өзара қарым-қатынас арқылы байланады, соның арқасында сұранымды орындау кезінде бірнеше кестелерден деректерді біріктіру мүмкіндігі қамтамасыз етіледі.

SQL MySQL жүйесінің бір бөлігі ретінде құрылымдық сұраныстар тілі ретінде сипаттауға болады және деректер қорына қол жеткізу үшін қолданылатын ең көп таралған стандартты тіл.

MySQL-ашық кодпен. Оны кез келген адам қолдана алады және өзгерте алады. Мұндай БҚ Internet арқылы алуға және тегін пайдалануға болады. Бұл жағдайда әрбір пайдаланушы бастапқы кодты оқып, оны өз қажеттіліктеріне сәйкес өзгерте алады.

MySQL бағдарламалық жасақтамасын пайдалану GPL (GNU General Public License) лицензиясымен реттеледі, http://www.gnu.org/licenses/, онда әр түрлі жағдайларда осы бағдарламалық қамтамасыз етумен не істеуге болатыны және не істеуге болмайтыны көрсетілген.

MySQL өте жылдам, сенімді және оңай. Егер сізге осы қасиеттер қажет болса, осы сервермен жұмыс істеуге тырысыңыз. MySQL сондай-ақ пайдаланушылармен тығыз байланыста әзірленген ыңғайлы мүмкіндіктерге ие.

MySQL сервері сол сәтте бар аналогтармен салыстырғанда жоғары жылдамдықты қамтамасыз ету мақсатында үлкен дерекқорларды басқару үшін әзірленген. Міне, бірнеше жыл бойы бұл сервер Жоғары талаптармен өнеркәсіптік пайдалану жағдайында табысты қолданылады. MySQL үнемі жетілдірілетініне қарамастан, ол бүгінгі күні пайдалы функциялардың кең спектрін қамтамасыз етеді. Өзінің қол жетімділігінің, жылдамдығы мен қауіпсіздігінің арқасында MySQL Internet деректер қорына қол жеткізу үшін өте қолайлы.

MySQL MySQL ДҚБЖ-ның техникалық мүмкіндіктері көп нүктелі SQL-серверді қамтитын клиент-сервер жүйесі болып табылады, ол деректер қорының әртүрлі есептеу машиналарын, сондай-ақ бірнеше түрлі клиенттік бағдарламалар мен кітапханаларды, әкімшілендіру құралдарын және бағдарламалық интерфейстердің кең спектрін (API) қолдауды қамтамасыз етеді. Біз сондай-ақ MySQL серверін көп ағынды кітапхана түрінде жеткіземіз,

оны пайдаланушы қосымшасына қосуға және жинақы, жылдам және оңай өнімді басқаруға болады. Сондай - ақ, MySQL үшін бағдарламалық қамтамасыз етудің үлкен саны бар, көп бөлігі-тегін.

MySQL екі бөліктен тұрады: серверлік және клиенттік.

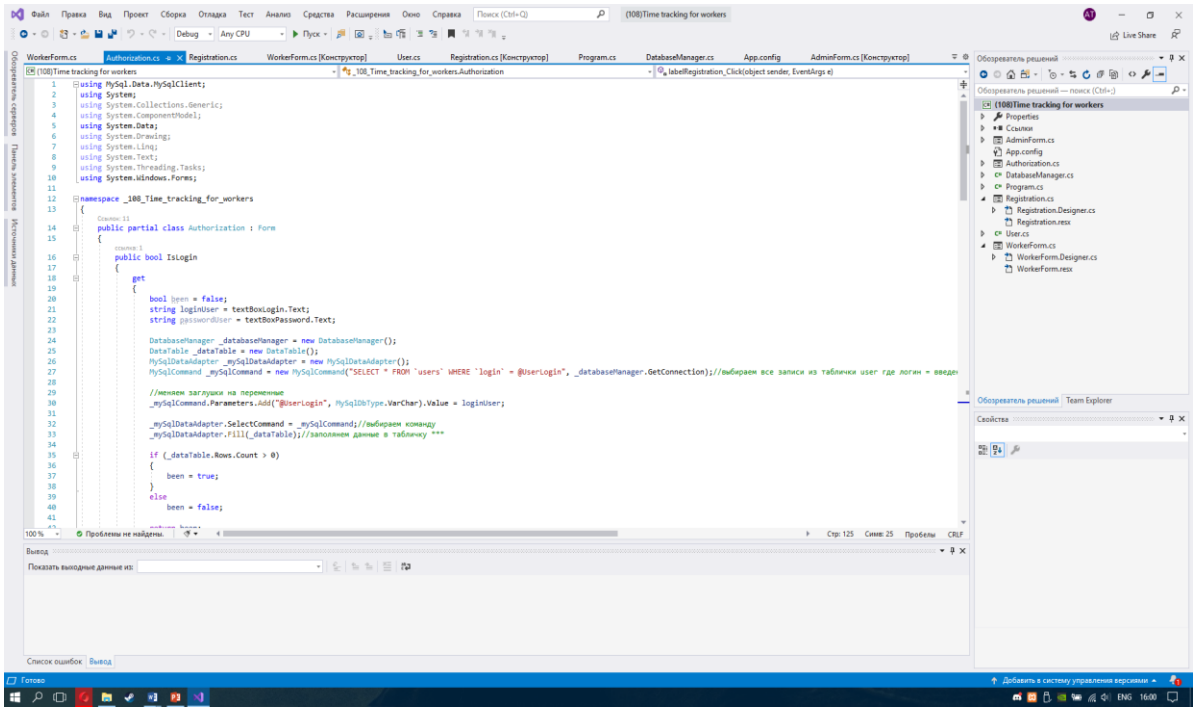
MySQL сервері үнемі компьютерде жұмыс істейді. Клиенттік бағдарламалар (мысалы, PHP скрипттері) MySQL SQL серверіне сокеттер механизмі арқылы (яғни желілік құралдардың көмегімен) сұраныстарды жібереді, сервер оларды өңдейді және нәтижені есте сақтайды. Яғни скрипт (клиент) деректер қорының серверінен қандай ақпарат алғысы келетінін көрсетеді. Содан кейін деректер қорының сервері клиентке (скриптке) жауап (нәтиже) жібереді. Неге әрқашан барлық нәтиже берілмейді? Өте қарапайым: нәтижелік деректер жиынтығының өлшемі тым үлкен болуы мүмкін және оны желі арқылы таратуға көп уақыт кетеді. Иә, сұраудың барлық қорытындысын бірден алу қажет болған кезде сирек (яғни сұрау салуды қанағаттандыратын барлық жазбалар). Мысалы, бізге қандай да бір шартты қанағаттандыратынын санауды немесе деректерді тек алғашқы 10 жазбаны таңдауды талап етуі мүмкін. Сокеттерді пайдалану механизмі клиент-сервер технологиясын білдіреді, ал бұл жүйеде арнайы бағдарлама – MySQL-сервер іске қосылуы тиіс, ол бағдарламалардан сұраныстарды қабылдайды және өңдейді. Өйткені барлық жұмыс жүреді шындыққа бір машинада, үстеме шығындар бойынша желілік құралдармен шамалы (орнату және қолдау қосылыстар MySQL – серверімен өнеркәсіпке өте арзан).

MySQL құрылымы үш деңгейлі: деректер қоры – кесте – жазбалар. Деректер қоры мен MySQL кестелері физикалық түрде frm, MYD, MYI кеңейтулері бар файлдармен ұсынылады. Логикалық-кесте жазбалардың жиынтығы болып табылады. Жазбалар-түрлі өрістердің жиынтығы. MySQL деректер қорының атауы жүйе шегінде бірегей, ал кестелер - деректер қоры шегінде, ал өрістер - кесте шегінде. Бір MySQL сервері бірден бірнеше дерекқорларды қолдай алады, оларға кіру логині мен пароль арқылы шектеле алады. Бұл логин мен құпия сөзді біле отырып, нақты деректер қорымен жұмыс істеуге болады. Мысалы, онда кестені жасауға немесе жоюға, жазбаларды қосуға және т.б. болады.

### 1.4.4 Microsoft Visual Studio бағдарламалық жасақтама жасау құралы

Microsoft Visual Studio – бұл C, C ++, VB.NET, C #, F #, JavaScript, Python сияқты танымал бағдарламалау тілдеріне қолдау көрсететін толық функционалды интеграцияланған даму ортасы (IDE).

Visual Studio функционалдығы бағдарламалық жасақтаманың дамуының барлық кезеңдерін қамтиды, кодты жазудың заманауи құралдарымен, GUI-дің дизайнымен, қосымшаларды құру, күйін келтіру және тестілеудің қазіргі заманғы құралдарымен қамтамасыз етеді. Visual Studio мүмкіндіктерін қажетті кеңейтімдерді қосу арқылы жақсартуға болады.



1.1 сурет – Visual Studio Code Editor синтаксисті бөлектеу

Visual Studio Code Editor синтаксисті бөлектеу, код үзінділерін қою және құрылым мен байланысты функцияларды көрсетуді қолдайды. Intelli AdSense технологиясы - терген кезде кодты толтыру жұмысты едәуір жылдамдатуға көмектеседі.

Visual Studio бағдарламасының кіріктірілген түзеткіші бастапқы кодтағы қателерді табу және түзету үшін қолданылады, оның ішінде аппараттық құралдардың деңгейі төмен. Диагностикалық құралдар кодтың сапасын өнімділік және жадты пайдалану тұрғысынан бағалауға мүмкіндік береді.Visual Studio Form Designer графикалық интерфейсі бар бағдарламаларды әзірлеу үшін қажет, ол болашақ қосымшаның сыртқы түрін және интерфейстің әр элементін жобалауға көмектеседі.

Сонымен, Visual Studio интерфейстердің жұмысын, блокты және жүктемені тексеру тұрғысынан қолданбалы тестілеуді автоматтандыруға арналған құралдар жиынтығын ұсынады.

Командалық жобалар үшін Visual Studio нақты уақыт режимінде кодтың кез-келген бөлігін бірлесіп өңдеуге және күйге келтіруге, сондай-ақ Team Foundation немесе Git нұсқаларын басқару жүйесі ретінде пайдалануға мүмкіндік беретін командалық жұмысты қолдауды ұсынады.

Microsoft Visual Studio-мен байланысты негізгі файл кеңейтімі - бұл SLN - Visual Studio Solution File, ашылған кезде бағдарламалық жасақтама шешіміне қатысты барлық мәліметтер мен жобалар бағдарламаға жүктеледі.

### 1.4.5 C# бағдарламалау тілі

C# − бұл объектіге бағытталған тіл, бірақ сонымен қатар компоненттік-бағытталған бағдарламалауды қолдайды. Заманауи қосымшалардың дамуы жеке функционалдылықты жүзеге асыратын оқшауланған және өзін-өзі сипаттайтын пакеттер түрінде бағдарламалық компоненттерді құруға барған сайын күш салуда. Мұндай компоненттердің басты ерекшелігі - олар қасиеттері, әдістері мен оқиғалары бар бағдарламалау моделін ұсынады. Олар компонент туралы декларативті ақпаратты беретін атрибуттарға ие. Олар өздерінің жеке құжаттарын қамтиды. C# жұмыс тұжырымдамасын тікелей қолдайтын тілдік конструкциялар ұсынады. Осыған байланысты C# бағдарламалық жасақтама компоненттерін құруға және қолдануға жарамды.

Мұнда қосымшалардың сенімділігі мен тұрақтылығын қамтамасыз ететін C# тілінің бірнеше ерекшеліктері берілген. Қоқыс жинау қол жетімді емес нысандардың жадын автоматты түрде босатады. Ерекшеліктер қателіктерді анықтауға және қалпына келтіруге құрылымды және кеңейтілген тәсіл ұсынады. Тілдің типтік қауіпсіз құрылымы іске асырылмаған айнымалылардан, индексті массивтерден олардың шекарасынан оқуға немесе тексерілмеген типті түрлендірулерді оқуға мүмкіндік бермейді.

C# бір типті жүйе бар. Барлық C# типтері, оның ішінде int және double сияқты қарапайым типтер бірдей түбірлік нысан түрінен өтеді. Осылайша, барлық типтер операциялардың жалпы жиынтығын пайдаланады, және кез-келген түрдегі мәндер ұқсас түрде сақталуы, берілуі және өңделуі мүмкін. Сонымен қатар, C# анықтамалық типтер мен мән түрлерін қолдайды, бұл сізге объектілерді жадты динамикалық түрде бөлуге, сонымен қатар қарапайым құрылымдарды стекке сақтауға мүмкіндік береді.

C# бағдарламалары мен кітапханалардың әрі қарай дамытылуымен үйлесімділігін қамтамасыз ету үшін C# құру кезінде нұсқаны басқаруға көп көңіл бөлінді. Көптеген бағдарламалау тілдері бұл сұрақты елемейді. Нәтижесінде, осы тілдердегі бағдарламалар тәуелді кітапханалардың жаңа нұсқалары шыққан кездегіден гөрі жиі бұзылады. Нұсқаны басқаруға қатысты мәселелер жеке виртуалды және қайта жазылатын модификаторлар, әдіс артық жүктемелерін шешу ережелері және интерфейс мүшелерін анық мәлімдейтін қолдау сияқты # даму аспектілеріне айтарлықтай әсер етті.

C# соңғы нұсқаларында әртүрлі бағдарламалау парадигмалары қолданылған. C# функционалды бағдарламалау әдістерін қолдайтын функцияларды қамтиды, мысалы ламбда өрнектері. Басқа жаңа мүмкіндіктер мәліметтер мен алгоритмдердің бөлінуін қолдайды, мысалы, үлгіні сәйкестендіру.

C# -та ұйымдық құрылымның негізгі ұғымдары - бағдарламалар, атаулар кеңістігі, түрлері, мүшелері және жиналыстары. C# бағдарламасы бір немесе бірнеше файлдан тұрады. Бағдарлама құрамында мүшелері бар типтерді жариялайды. Бұл типтерді аттар кеңістігінде ұйымдастыруға болады. Түрлердің мысалдары - сыныптар мен интерфейстер. Мүшелерге өрістер, әдістер, қасиеттер және оқиғалар кіреді. Компиляция кезінде C# бағдарламалары жинақ түрінде жинақталады. Жиын - бұл, әдетте .exe немесе

.dll кеңейтімі бар файл, егер ол сәйкесінше қосымшаны немесе кітапхананы қолданса.

C# -та екі тип бар: сілтеме түрлері және мән түрлері. Типтік шамалардың айнымалы мәндері тікелей деректерді қамтиды, ал сілтеме түрлерінің айнымалылары объектілер деп аталатын қажетті мәліметтерге сілтемелерді сақтайды. Анықтамалық типтің екі айнымалы мәні бірдей объектіге сілтеме жасай алады, сондықтан бір айнымалыдағы әрекеттер басқа айнымалыға сілтеме жасайтын объектіге әсер етуі мүмкін. Әрбір айнымалы мәнде мәліметтердің өзіндік көшірмесі болады, ал бір айнымалыға арналған амалдар екіншісіне әсер ете алмайды (айнымалы параметрлерден басқа). C# мән түрлері қарапайым түрлерге, санау түрлеріне, құрылым түрлеріне және бос емес түрлерге бөлінеді. C# тіліндегі анықтамалық типтер сынып типіне, интерфейс типтеріне, массив типтеріне және өкіл түрлеріне бөлінеді.

C# − бұл қатты терілген тіл. Әр айнымалы және тұрақты мәннің әр өрнек сияқты типі болады, оның мәні есептеу болып табылады. Әр әдістің қолтаңбасы әрбір енгізу параметріне және қайтарылатын мәнге арналған түрді белгілейді. .NET класс кітапханасы файлдық жүйе, желілік қосылымдар, коллекциялар, объектілер массивтері және даталар сияқты логикалық конструкциялардың сан алуан түрін білдіретін кірістірілген сандық түрлердің жиынтығын, сонымен қатар одан да күрделі типтерді анықтайды. Әдеттегі C# бағдарламасы осы сынып кітапханасынан және белгілі бір қосымшаның бірегей тұжырымдамаларын модельдейтін типтерден тұрады.

### 1.4.6 phpMyAdmin веб-интерфейсті ұсынатын ашық бастапқы бағдарлама

PhpMyAdmin браузері арқылы MySQL серверін басқаруға және SQL командаларын орындауға ғана емес, сонымен қатар кестелер мен дерекқорлардың мазмұнын қарауға мүмкіндік береді. Бағдарлама веб-әзірлеушілер арасында өте танымал, себебі SQL командаларын тікелей енгізбестен MySQL ДҚБЖ-ны басқаруға мүмкіндік береді.

Бағдарлама GNU General Public лицензиясы бойынша таратылады, сондықтан көптеген басқа әзірлеушілер оны өздерінің дамуымен біріктіреді, мысалы XAMPP, Denwer, AppServ, Open Server.

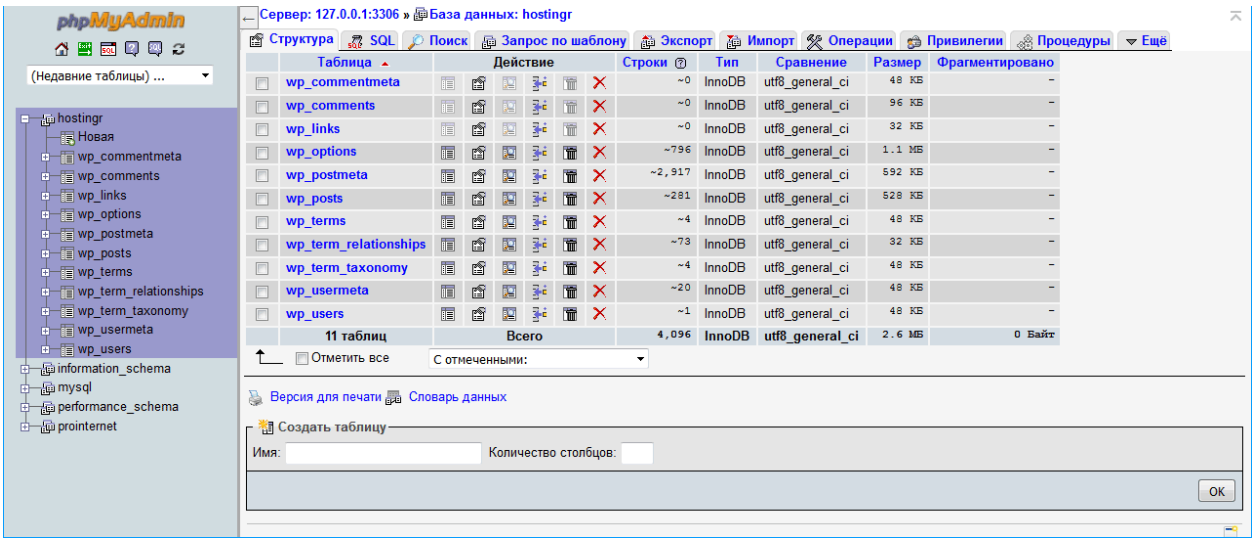
Бастамашы 1997 жылы пайда болған MySQL-Webadmin ұқсас қосымшасына негізделген неміс жасаушысы Тобиас Рацчилер (Tobias Ratschiller) және phpPgAdmin-де негізделген.

2001 жылдан бастап phpMyAdmin жобасын негізін қалаушы Тобиас Рацчилерден Марк Делисл қабылдады, Оливье Мюллер мен Loïc Chapeaux бастап phpMyAdmin жаңа дәуірі басталды.

Бүгінгі таңда Марк Делислис жоба әкімшісі болып табылады және phpMyAdmin-ге жазбаша код түрінде ең көп үлес қосады. Жобаның негізгі көмекшілері - Михал Шиха, Себастьян Мендель және Герман ван Ринк. Дерекқорды басқару бағдарламасы шолғыш терезесінде ашылады. Оның

интерфейсі қысқа және түсінікті, қосымша нұсқауларсыз. Орысша жазылған нұсқасы бар. PHPMyAdmin интерфейсі екі бөлікке бөлінген терезеден тұрады.

Сол жақта тар баған орналасқан, онда мәліметтер базасының тізімі және олардың әрқайсысының ағаш құрылымы көрсетілген. Оң жақта негізгі ұяшықтардың мазмұны, командалар мен қызмет көрсету батырмаларының мәзірі көрсетілетін жұмыс терезесі орналасқан.




1.1 сурет - PHPMyAdmin интерфейсі

Ұяшықтармен және олардың құрамымен жасалатын барлық операциялар - қарау, көшіру, жою, қою бір шертумен орындалады. Көбінесе веб-әзірлеушілер мен сайт әкімшілері сайтты бір хосттан екіншісіне ауыстыру үшін дереу деректер қорын құру (көшіру) мүмкіндігін пайдаланады. Қаласаңыз, оны бірден мұрағатталған түрде алуға болады.

PHPMyAdmin-нің тағы бір қызықты ерекшелігі - сайттың әкімшілік бөлігіне кіру үшін парольді қалпына келтіру. Пайдаланушылар ұяшығына немесе соған ұқсас бөлімге өтіп, өзіңіздің есептік жазбаңызға қажетті логин мен парольді көшіріп алыңыз. Бұл бағдарлама сәтсіздікке ұшыраған жағдайда сайттың әкімшілік бөлігін сәтті ауыстыра алады.

## 2 Жұмыс уақытын есептеу бағдарламасын жобалау

Бұл бағдарламалық қамтамасыз ету компания қызметкерлерінің жұмыс уақытын есепке алатын басты беттен (пайдаланушылық бөлім) және басқару панелінен (администраторлық бөлім) тұрады.

Қызметкерлердің жұмыс уақытын есепке алу үшін мен қызметкерлердің кіру және кету уақытын жазатын жеңіл бағдарлама жасауды ұйғардым. Бағдарлама коммерциялық емес болғандықтан, оны мүмкіндігінше оңай пайдаландым және қосымша жабдықтар мен құрылғыларды қажет етпедім.

### 2.1 Пән саласын модельдеу

Пәндік саланы модельдеу модельдің статикалық бөлігінің негізі болып табылады. Пәндік сала моделін құру нақты әлемде бар абстракцияларды, яғни жүйеде кездесетін концептуалды объектілерді анықтаудан басталады. Жалпы объектілі модельдеудің және статикалық модельдеудің негізі, атап айтқанда пәндік аймақтан осы абстракциялардың моделін құру болып табылады. Пәндік аймақ көптеген фрагменттермен ұсынылған, мысалы, кітап дүкені дүкендер жиынтығы ретінде ұсынылуы мүмкін - сауда алаңы, қойма, басқару аппараты, бухгалтерия және т.б. Пәндік аймақтың әр фрагменті көптеген нысандар мен объектілерді пайдаланатын процестермен, сонымен қатар көптеген пайдаланушылармен, тақырыптық аймаққа әртүрлі көзқарастармен сипатталады. Тұсаукесер деп аталатын мұндай көрініс осы қолданушының көзқарасы бойынша тақырып аймағының негізгі және маңызды аспектілерін қамтиды, ал маңызды емес аспектілер қарастырудан шығарылады. Пәндік саланың моделі болашақта жүйенің прецеденттерін анықтау және сипаттау үшін қолданылатын терминдер сөздігін білдіреді. Пәндік саладағы объектілерді анықтау барысында олардың арасында қандай байланыстар бар екенін анықтау қажет. Пәндік аймақтың моделін моделдеу кезінде "ішкі сыртқа" жобалау әдістемесі қолданылады. Осылайша, прецеденттер немесе жүйенің динамикалық бөлігі анықталған кезде сыртқы ішке қарай, ал статикалық модель құру кезінде – ішкі сыртқа қарай қозғалады.

Деректер базасы тек ақпаратты ғана емес, сонымен қатар деректер құрылымын сипаттайтын репозиторий. Мұндай сипаттама әдетте деректер сөздігі деп аталады, ал сипаттама элементтерінің өздері метамәліметтер, яғни деректер туралы мәліметтер деп аталады. Деректер сөздігі дерекқорды ұйымдастырудың негізгі қағидатын - қолданылған бағдарламалық жасақтамаға тәуелсіздік береді.

### 2.1.1. Деректер қорын модельдеу

Деректерді модельдеу - мәліметтер базасын құрудың маңызды кезеңі. Деректер моделі бизнес туралы ақпаратты ұсынады. Егер мәліметтер моделінде ақаулар болса, онда ақаулар мәліметтер базасының өзінде де, оған кіретін барлық бағдарламаларда да пайда болады. Деректер үлгісі және мәліметтер базасы өзінің икемділігі мен кеңею перспективаларын қамтамасыз ететін етіп жасалуы керек.

Деректер базасын жобалау - бұл мәліметтер базасының схемасын құру және қажетті тұтастық шектеулерін анықтау процесі. Деректер базасын құрудың негізгі кезеңдері:

- Тұжырымдамалық (инфологиялық) дизайн

пәндік аймақтың семантикалық моделін, яғни абстракцияның жоғарғы деңгейінің ақпараттық моделін құру. Мұндай модель ДББЖ-ге және деректер моделіне назар аудармай жасалады. «Семантикалық модель», «тұжырымдамалық модель» және «инфологиялық модель» терминдері синонимдер болып табылады. Сонымен қатар, осы контексте «мәліметтер базасының моделі» және «домендік модель» сөздерін (мысалы, «дерекқордың тұжырымдамалық моделі» және «тұжырымдамалық домендік модель») қолдануға болады, өйткені мұндай модель шындықтың бейнесі де, бейнесі де болып табылады Бұл шындыққа арналған дизайн мәліметтер базасы.


2.1 сурет – Тұжырымдамалық (Концептуальная) схеманың мысалы

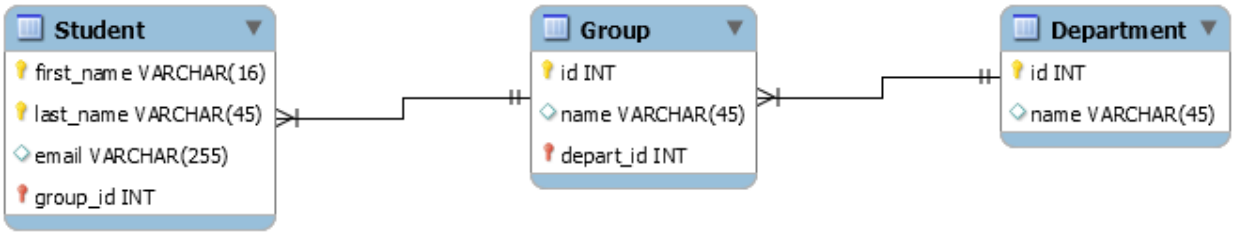
Дерекқордың тұжырымдамалық моделінің нақты нысаны мен мазмұнын осы үшін таңдалған ресми аппарат анықтайды. Әдетте ER диаграммалары сияқты графикалық белгілер қолданылады.

Көбінесе дерекқордың тұжырымдамалық моделі мыналарды қамтиды: ақпараттық объектілердің сипаттамасы немесе пәндік аймақ түсінігі және олардың арасындағы қатынастар;

Тұтастық шектеулерінің сипаттамасы, яғни дұрыс деректер мәндеріне қойылатын талаптар және олардың арасындағы қатынастар.

- Логикалық (каталогтық) дизайн - нақты деректер үлгісіне негізделген мәліметтер базасының схемасын құру, мысалы, реляциялық деректер моделі. Реляциялық деректер моделі үшін каталогтық модель дегеніміз - әдетте бастапқы кілттермен, сондай-ақ шетелдік кілттер болып табылатын

қатынастар арасындағы қатынастардың жиынтығы.




2.2 сурет – Логикалық схеманың реляциялық мәліметтер моделінің мысалы

Тұжырымдамалық модельді логикалық модельге айналдыру, әдетте, ресми ережелерге сәйкес жүзеге асырылады. Бұл кезең негізінен автоматтандырылуы мүмкін.

Логикалық дизайн кезеңінде белгілі бір деректер моделінің ерекшеліктері ескеріледі, бірақ нақты ДҚБЖ ерекшеліктері ескерілмеуі мүмкін.

Физикалық дизайн - нақты ДҚБЖ үшін мәліметтер базасының схемасын құру. Белгілі бір ДҚБЖ-нің ерекшелігі дерекқор нысандарының атына шектеулер, қолдау көрсетілетін мәліметтер түрлеріне шектеулерді қамтуы мүмкін. Сонымен қатар, физикалық дизайн кезінде нақты ДҚБЖ ерекшелігі физикалық сақтау ортасына қатысты шешімдерді таңдауды (дискіні басқару әдістерін таңдау, мәліметтер базасын файлдар мен құрылғыларға бөлу, деректерге қол жеткізу әдістері), индекстер құру және т.б.

SQL-де жоғарыда көрсетілген логикалық диаграмманың физикалық дизайнының нәтижесі келесі сценарий болуы мүмкін:

```sql
CREATE TABLE IF NOT EXISTS Department ( -- Факультет
  id INT NOT NULL,
  name VARCHAR(45),
  PRIMARY KEY (id)
);

CREATE TABLE IF NOT EXISTS Group (
  id INT NOT NULL,
  name VARCHAR(45) ,
  depart_id INT NOT NULL,
  UNIQUE INDEX depart_id_UNIQUE (depart_id ASC),
  PRIMARY KEY (id, depart_id),
  CONSTRAINT depart_fk
    FOREIGN KEY (depart_id)
    REFERENCES Department (id)
);

CREATE TABLE IF NOT EXISTS Student (
  first_name VARCHAR(16) NOT NULL,
  last_name VARCHAR(45) NOT NULL,
  email VARCHAR(255),
  group_id INT NOT NULL,
  PRIMARY KEY (last_name, first_name, group_id),
  INDEX group_fk_idx (group_id ASC),
  CONSTRAINT group_fk
    FOREIGN KEY (group_id) REFERENCES Group (id)
);
```

2.1 сурет - SQL скриптінің мысалы

### 2.1.2 Сілтеме модельдері

1976 жылы П.Чен [1] ұсынған «Субъектілік қатынастар» моделі немесе ER моделі семантикалық (тұжырымдамалық, инфологиялық) домендік модельдер класының ең танымал өкілі болып табылады. ER моделі әдетте П.Ченнің бастапқы жазбасын, ER диаграммасы деп аталатын немесе басқа графикалық белгілерді (Crow's Foot, Information Engineering және т.б.) қолдана отырып, графикалық түрде ұсынылады.

ER-модельдерінің негізгі артықшылықтары:

- көріну;
- модельдер көптеген объектілер мен атрибуттармен мәліметтер базасын жасауға мүмкіндік береді;

- ER-модельдер көптеген автоматтандырылған жобалау жүйелерінде енгізілген (мысалы, ERWin).

ER-модельдерінің негізгі элементтері:

- объектілер (субъектілер);
- объектілердің атрибуттары;
- объектілер арасындағы байланыс.

Нысан - бұл атрибуттары бар объектінің домені.

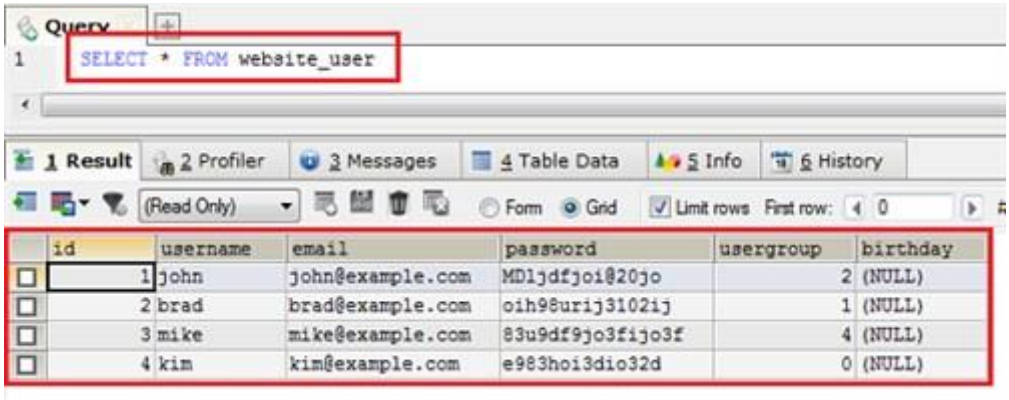
Субъектілердің арасындағы қатынастар мыналармен сипатталады:

- байланыс түрі (1: 1, 1: N, N: M);
- туыстылық класы. Сынып қажет және міндетті емес. Егер ұйымның әрбір данасы қарым-қатынасқа қатысатын болса, онда мүшелік класы міндетті болып табылады, әйтпесе ол міндетті емес.

### 2.1.3 Құрылымдық сұрау тілі

Ондағы ақпаратты сақтау және қажет болған жағдайда осы ақпаратты алу үшін мәліметтер базасы жасалады. Бұл деректерді дерекқорға (INSERT) кіргізіп, кіргізе білуіміз керек және дерекқордан (SELECT) ақпараттарды таңдай аламыз дегенді білдіреді.

Осы мақсаттар үшін мәліметтер қорының сұранысы ойлап табылды және құрылымдалған сұрау тілі немесе SQL деп аталды. Деректерді енгізу операциялары (INSERT) және оларды таңдау (SELECT) - осы тілдің өзі. Төменде деректерді және оның нәтижесін алуға арналған сұраудың мысалы келтірілген.

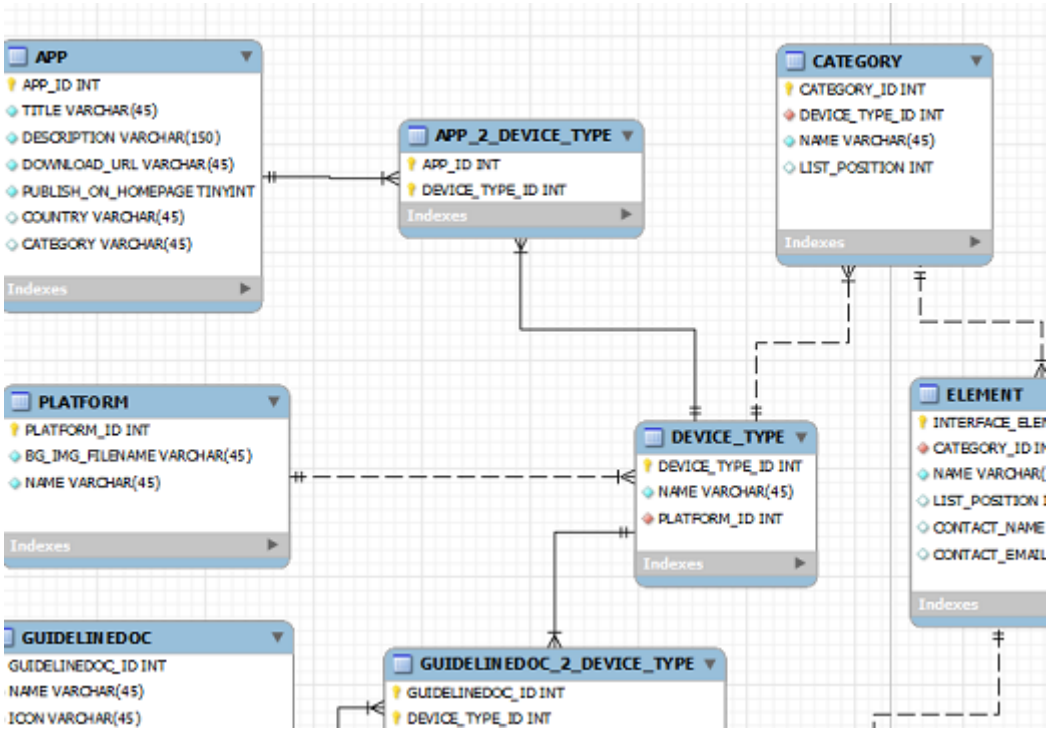



2.3 сурет – Деректердің нәтижесін алуға арналған сұрау мысалы

SQL - әңгімелеудің үлкен тақырыбы және оны қарау осы нұсқаулықтың шеңберіне кірмейді. Бұл мақала мәліметтер базасын құру процесінің презентациясына қатаң бағытталған. Кейінірек бөлек нұсқаулықта SQL негіздерін қарастырамын.

### 2.1.4 Реляциялық модель.

Реляциялық модель дегеніміз - кестелердегі мәліметтерді қалай ұйымдастыруды және осы кестелер арасындағы қатынасты қалай анықтайтындығын сипаттайтын модель.




2.4 сурет – Реляциялық модельгк мысал

Реляциялық модель ережелері ақпаратты кестелерде қалай құру керектігін және кестелер бір-бірімен қалай байланысты болатындығын белгілейді. Сайып келгенде, нәтижені суретте көрсетілгендей мәліметтер базасының диаграммасы немесе дәлірек айтқанда, ұйым-қатынастар диаграммасы түрінде ұсынуға болады (Мысал MySQL Workbench-тен алынған).

### 2.1.5 Реляциялық мәліметтер базасының сипаттамасы.

Реляционные базы данных разработаны для быстрого сохранения и получения больших объемов информации. Ниже приведены некоторые характеристики реляционных баз данных и реляционной модели данных.

Кілттерді қолдану. Кестедегі әр жолды бастапқы кілт деп аталатын ерекше «кілт» анықтайды. Көбіне бастапқы кілт автоматты түрде көбейтілетін (авто-өсіру) нөмір болып табылады (1,2,3,4, т.б.). Әр түрлі кестелердегі мәліметтерді пернелер көмегімен байланыстыруға болады. Бір кестенің бастапқы кілт мәндерін басқа кестенің жолдарына (жазбаларына) қосуға болады, осылайша осы жазбаларды бір-бірімен байланыстырады.

Жаңа кесте құру мысалында, қажетті өрістерді тізімдеген кезде, бірінші өріс - ID INT AUTO_INCREMENT PRIMARY KEY.

Бұл өріс бастапқы кілт деп аталады. Әрбір кестеде бастапқы кілт жасағаныңызға көз жеткізіңіз.

Бастапқы кілт - жазбаның бірегей идентификаторы сақталатын арнайы өріс. Бұл бағдарламашы мен деректер базасында оны оқу, жаңарту немесе жою үшін бір нақты жазбаға әрқашан нақты сілтеме жасау мүмкіндігі болуы үшін қажет.

Егер сіз өрісті бастапқы кілтпен тағайындасаңыз, мәліметтер базасы осы өрістегі мәннің кестеде қайталанбауын қамтамасыз етеді.

Егер сіз сонымен қатар AUTO_INCREMENT атрибутын қоссаңыз, MySQL жаңа жазбаларды қосқанда бұл өрісті өзі толтырады. AUTO_INCREMENT есептегіш рөлін атқарады - кестеде әрбір жаңа жазба максималды мәннен үлкен мән алады.

Құрылымдық сұрау тілін (SQL) пайдаланып, кілтпен байланысқан әр түрлі кестелердегі деректерді бір уақытта таңдауға болады. Мысалы, сіз тұтынушылар кестесінен идентификаторы (id) 3 (Майк) бар пайдаланушыға тиесілі тапсырыстар кестесінен барлық тапсырыстарды таңдайтын сұрау жасай аласыз.

| | id | username | email | password | usergroup |
|---|---|---|---|---|---|
| ☐ | 1 | john | john@example.com | MDljdfjoi@2( | 2 |
| ☐ | 2 | brad | brad@example.com | oih98urij31( | 1 |
| ☐ | 3 | mike | mike@example.com | 83u9df9jo3fj | 4 |
| ☐ | 4 | kim | kim@example.com | e983hoi3dio3 | 1 |

2.5Сурет - кестеге кілттердің қолдану мысалы

Кестеде id бағанында бастапқы кілт бар. Әр енгізілімде ерекше бастапқы кілт болады, көбінесе нөмір. Пайдаланушы тобының бағанында шетелдік кілт болады. Аты-жөні бойынша, бұл пайдаланушылар топтары бар кестеге қатысты болуы мүмкін.

Реляциялық деректер моделінің ережелерін ескере отырып жасалған мәліметтер базасының жобасында пайдаланушы аты сияқты ақпараттың әр бөлігі тек бір жерде сақталады. Бұл бірнеше жерде деректермен жұмыс істеу қажеттілігін жояды. Деректердің қайталануы деректердің артықтығы деп аталады және жақсы мәліметтер қорын жасақтауда оны болдырмауға болады.

Реляциялық дерекқорды пайдаланып, бағанда қандай деректер түрін сақтауға болатынын анықтауға болады. Сіз бүтін сандардан, ондық сандардан, мәтіннің кішігірім бөліктерінен, мәтіннің үлкен бөліктерінен, даталардан және т.б. тұратын өрісті жасай аласыз.

| | Field Name | Datatype | Len | Def |
|---|---|---|---|---|
| | id | int | 11 | |
| | username | varchar | 50 | |
| | email | varchar | 50 | |
| | password | varchar | 100 | |
| | usergroup | int | 10 | |
| | birthday | date | | |
| * | | | | |

2.6 Сурет – Кестеге еңгізген айнымалылардың типтері

Деректер кестесін құрған кезде әр баған үшін деректер түрін бересіз. Мысалы, varchar - бұл кішігірім мәтіндік үзінділер үшін мәліметтер типі, ең көбі 255 таңбадан тұрады, ал int - сандар.

Деректер түрлеріне қосымша, RDBMS сізге енгізуге болатын деректерді одан әрі шектеуге мүмкіндік береді. Мысалы, ұзындығын шектеңіз немесе осы бағандағы жазбалар мәнінің бірегейлігін мәжбүрлеңіз. Соңғы шектеу көбінесе пайдаланушының кіру аттары (логиндері) немесе электрондық пошта мекенжайлары бар өрістер үшін қолданылады.

Бұл шектеулер деректердің тұтастығын бақылауға мүмкіндік береді және келесідей жағдайлардың алдын алады:

- жолды (мәтінді) көру керек жолға енгізіңіз
- аймақтың индексін осы индекстің ұзындығына жүз таңбамен енгізіңіз
- бірдей атаумен пайдаланушылар құру
- бірдей электрондық пошта мекенжайы бар пайдаланушыларды құру
- туған күн (күн) жолына салмақты (нөмірді)

Өрістердің қасиеттерін реттеу, кестелерді бір-бірімен байланыстыру және шектеулер қою арқылы сіз деректеріңіздің сенімділігін арттыра аласыз.

Көптеген RDBMS белгілі бір пайдаланушыларға нақты құқықтар тағайындауға мүмкіндік беретін рұқсат параметрлерін ұсынады. Пайдаланушыға рұқсат етілуі немесе тыйым салынуы мүмкін кейбір әрекеттер: SELECT (таңдау), INSERT (кірістіру), ЖОЮ (жою), ALTER (өзгерту), ЖАСАУ (жасау), т.б. Бұл құрылымдалған сұрау тілі (SQL) көмегімен орындалатын операциялар.

### 2.1.6 Құрылымды сұрау тілі (SQL).

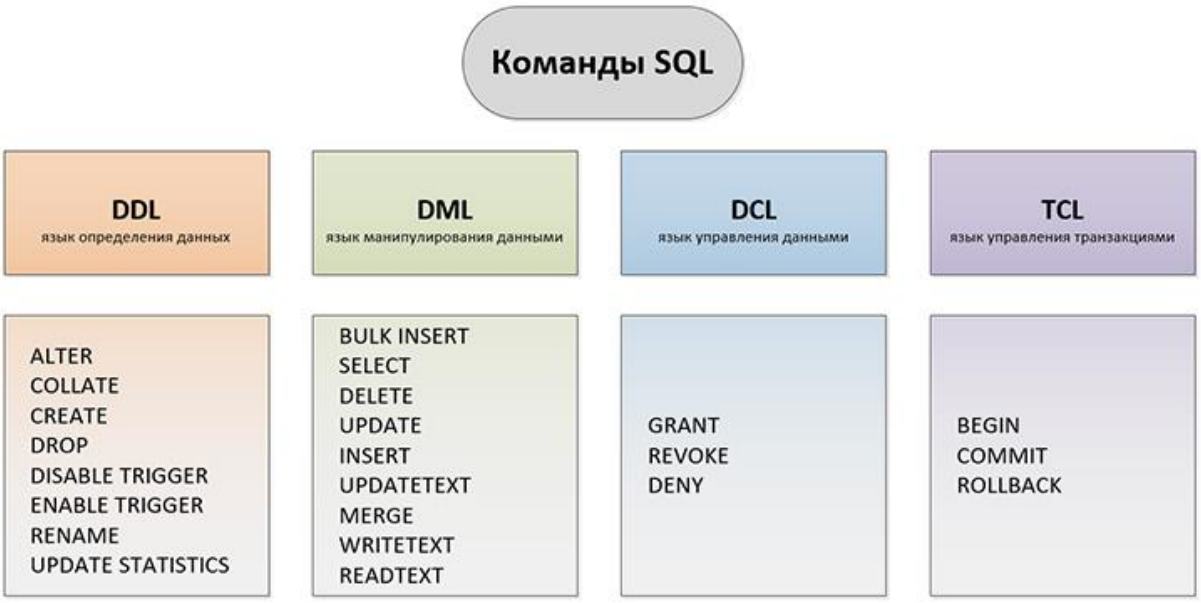
Деректер базасында белгілі бір операцияларды орындау үшін, мысалы, мәліметтерді сақтау, таңдау, өзгерту, құрылымдалған сұрау тілі (SQL) қолданылады. SQL түсіну оңай және мүмкіндік береді және SQL JOIN нұсқауын қолдана отырып, бірнеше кестелерден деректерді алу сияқты жинақталған таңдау. Жоғарыда айтылғандай, SQL осы нұсқаулықта талқыланбайды. Мен мәліметтер базасын жобалауға назар аударамын. SQL сұраулардың келесі түрлері бөлінеді:

DDL (Data Definition Language) - деректерді анықтайтын тіл. DDL сұрауларының міндеті - мәліметтер базасын және оның құрылымын сипаттау. Осындай түрдегі сұраныстар дерекқорға әртүрлі деректерді орналастырудың ережелерін белгілейді.

DML (Data Manipulation Language) - деректерді басқару тілі. Осы типтегі сұрауларға кейбір мәліметтерді басқару тікелей орындалатын әртүрлі командалар кіреді. DML сұраныстары бұрын енгізілген мәліметтерге өзгерістер енгізу үшін, дерекқордан мәліметтер алу, оларды сақтау, әртүрлі жазбаларды жаңарту және дерекқордан жою үшін қажет. DML қоңырауларының элементтеріне SQL нұсқамаларының көп бөлігі кіреді.

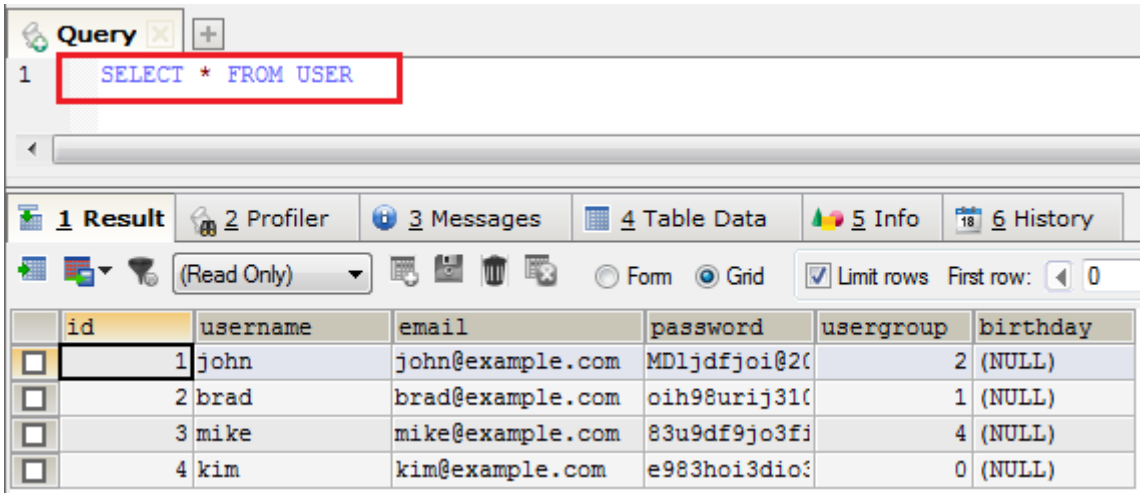
DCL (Data Control Language) - бұл деректерді басқару тілі. Оған рұқсаттар, құқықтар және басқа ДҚБЖ параметрлеріне қатысты сұраулар мен командалар кіреді.

TCL (Transaction Control Language) - транзакцияны басқару тілі. Құрылымның бұл түрі DML сұраныстарының көмегімен жасалған өзгерістерді басқару үшін қолданылады. TCL құрылымдары DML сұрауларын транзакциялар жиынтығына біріктіруге мүмкіндік береді.




2.7 Сурет - SQL сұрауларының түрлері бойынша негізгі түрлері

Дерекқорды қалай жасағаныңыз дерекқордан деректерді алу үшін толтыру қажет болатын сұрауларға тікелей әсер етеді. Бұл сіздің базаңыз қандай болуы керек екендігі туралы ойланудың тағы бір себебі. Жақсы жасақталған мәліметтер базасында сіздің сұрауларыңыз неғұрлым таза және қарапайым бола алады.




2.8Суерт – Деректер қорында сұраныстардың мысалы

Реляциялық деректер моделі стандартты болып табылады. Реляциялық деректер моделінің ережелерін сақтай отырып, сіздің деректеріңізді басқа RDBMS-ге салыстырмалы түрде оңай беруге болатындығына сенімді бола аласыз.

Жоғарыда айтылғандай, дерекқорды жобалау - бұл деректерді анықтау, оның байланысы және осы мәселені шешудің нәтижелерін қағазға (немесе компьютерлік бағдарламада) қою. Деректер базасын құру сіз оны құру үшін пайдаланғыңыз келетін РДБЖ-ға тәуелсіз.

### 2.1.7 C # оқиғаларды дискретті модельдеу

DES жүйені немесе процесті уақыт ішінде жеке оқиғалардың реттелген реті ретінде, яғни бір оқиғадан бастап келесі оқиғаның сәтіне дейін модельдейді. Сондықтан, DES модельдеу кезінде уақыт нақты уақытқа қарағанда әлдеқайда қысқа.

DES модельдеуін жасау кезінде ескерілетін алты негізгі элемент бар.

- Нысандар нақты жүйенің элементтерін ұсынады. Олардың қасиеттері бар, олар оқиғаларға жауап береді, ресурстарды пайдаланады, сонымен қатар уақыт өте келе кезекке кіреді және кетеді. Жоғарыда аталған ұшу және қону сценарийлерінде объектілер әуе кемелері болып табылады. Денсаулық сақтау жүйесінде науқастар немесе органдар болуы мүмкін. Қойма жүйесінде заттар қоймадағы тауарлар болып табылады. Нысандар бір-бірімен немесе жүйемен өзара әрекеттеседі және барлық модельдеу кезінде кез-келген уақытта құрылуы мүмкін деп болжанады.
- Қасиеттер Модельдеудің мүмкін болатын әр түрлі сценарийлердің жауабын анықтау үшін сақталатын әр объектінің нақты сипаттамалары (мөлшері, қону уақыты, еден, бағасы және т.б.); олардың мәні әртүрлі болуы мүмкін
- Оқиғалар жүйеде болуы мүмкін нәрсе, әсіресе объектілермен, мысалы, ұшақтың қонуы, қоймаға тауарларды жеткізу, белгілі бір аурудың пайда болуы және т.б. Оқиғалар кез-келген тәртіпте пайда болуы және қайта пайда болуы мүмкін.
- Ресурстар нысандарға қызмет көрсететін элементтер (айталық, аэродромға қону жолақтары, қоймадағы сақтау камералары және емханадағы дәрігерлер). Ресурс бос емес және кез-келген объектіге қажет болған кезде, бұл объект кезекке қойылып, ресурстың босатылуын күту керек.

- Кезектер объектілер ұйымдастырылған делдал құрамдастары кез-келген уақытта жұмыс істейтін кез-келген ресурстың шығарылуын күтеді. Кезектер максималды сыйымдылыққа ие және әртүрлі принциптерді қолдана алады: «бірінші-бірінші-шығу» (бірінші-бірінші-шығу, ФИФО), «соңғысы-бірінші-шығу» (соңғысы-бірінші, LIFO) немесе шығарылған кейбір өлшемдердің немесе басымдылықтардың негізі (аурудың өршуі, отын шығыны және т.б.).
- Уақыт Шынайы өмірдегідей, бұл модельдеуде маңызды рөл атқаратын фактор. Уақытты өлшеу үшін сағат модельдеудің басталуынан басталады; онда уақытты белгілі бір кезеңдерді (ұшу немесе келу уақыты, тасымалдау уақыты, белгілі бір белгілерді сақтау уақыты және т.б.) бақылау үшін пайдалануға болады. Мұндай қадағалау өте маңызды, өйткені бұл сізге келесі оқиғаның қашан басталатынын білуге мүмкіндік береді.

Модельдеу бағдарламалау өте күрделі болуы мүмкін болғандықтан, дамуды жеңілдету үшін модельдеу парадигмасының барлық талаптарын қамтитын тілдерді жасауға бірнеше әрекет жасалды. Осындай тілдердің бірі - 1960 жылдары Оле Йохан Оле Йохан мен Кристен Найгаард ойлап тапқан SIMULA және объектілі-бағытталған бағдарламалау (ООР) тұжырымдамасын енгізген алғашқы тіл - қазіргі парадигма. бағдарламалау. Қазіргі уақытта модельдеу жасау кезінде бағдарламашыларға қажет болатын пакеттерді, инфрақұрылымдарды немесе кітапханаларды құруға көп көңіл бөлінеді. Бұл кітапханаларды C #, C ++, Java немесе Python сияқты қарапайым тілдерден шақыру керек екендігі түсінікті.

«Оқиғаларды дискреттік бағдарламалаудың тілдерінің тарихы» мақаласында Nance DES-тің кез-келген бағдарламалау тіліне сәйкес келуі керек дегенде алты талапты ұсынды.

- кездейсоқ санды құру;
- біркелкі кездейсоқ шамадан басқа айнымалыларды қолдануға мүмкіндік беретін технологиялық трансформаторлар;
- объектілерді құруды, олармен айла-шарғы жасауды және оларды жоюды жеңілдететін тізімдерді өңдеу;
- модельдік мінез-құлықтың қысқаша сипаттамасын беретін статистикалық талдау;
- үлкен деректер жиынтығын ұсынуға көмектесетін және шешім қабылдауға ықпал ететін есептерді құру;
- уақыттың өту механизмі.

Барлық осы талаптарды C # -де қанағаттандыруға болады, сондықтан мен осы мақалада мен C # тобындағы популяцияны көбейту үшін оқиғалардың дискретті моделін енгіземін.

### 2.2 Интерфейс прототипі

Интерфейстің бұл прототипі көптеген корпоративтік компаниялар үшін қолайлы. Жұмыс орнына келген уақытта пайдаланушы жеке компьютерін қосып осы бағдарламада авторизацияланады. Егер пайдаланушы логині мен паролін дұрыс енгізсе, бағдарлама оның жұмыс басталу уақытын жазып алады. Жұмыс күні аяқталған кезде пайдаланушы бағдарламаны жабу кезінде бағдарламаны жабуы тиіс, ол пайдаланушының логинінен шығып, оның жұмыс аяқталған уақытын жазады, бағдарламаны мерзімінен бұрын жабу кезінде ол сондай-ақ қызметкер жұмысты қанша бұрын аяқтағанын көруге болады. Бұл бағдарламаның артықшылығы - оның қарапайым басқару элементтері.

Интерфейс кейбір функционалдылықты анықтай алатын анықтамалық типті - жүзеге асырусыз әдістер мен қасиеттер жиынтығын білдіреді. Содан кейін бұл функцияны осы интерфейстерді қолданатын класстар мен құрылымдар жүзеге асырады.

#### 2.2.1 Интерфейс түрін анықтау

Интерфейс кілт сөзі интерфейсті анықтау үшін қолданылады. Әдетте, C # тіліндегі интерфейс атаулары I бас әріптерінен басталады, мысалы, IComparable, IEnumerable (венгр деп аталатын), бірақ бұл міндетті талап емес, бірақ бағдарламалау стилі.

Интерфейс нені анықтай алады? Жалпы, келесі нысандар интерфейстерді анықтай алады:

- Әдістері;
- Қасиеттері;
- Индикаторлар;
- Оқиғалар;

Статикалық өрістер және тұрақтылар (C # 8.0 нұсқасынан бастап)

Алайда, интерфейстер статикалық емес айнымалыларды анықтай алмайды. Мысалы, барлық осы компоненттерді анықтайтын қарапайым интерфейс:

```
interface IMovable
{
    // константа
    const int minSpeed = 0;     // минимальная скорость
    // статическая переменная
    static int maxSpeed = 60;   // максимальная скорость
    // метод
    void Move();                // движение
    // свойство
    string Name { get; set; }   // название

    delegate void MoveHandler(string message);  // определение делегата для события
    // событие
    event MoveHandler MoveEvent;    // событие движения
}
```

2.9 Сурет – Қарапайым интерфейс мысалы

Бұл жағдайда қозғалмалы нысанды білдіретін жылжымалы интерфейс анықталады. Бұл интерфейс қозғалатын объектінің мүмкіндіктерін сипаттайтын әртүрлі компоненттерді қамтиды. Яғни, интерфейс қозғалатын объектіде болуы керек кейбір функцияларды сипаттайды.

Интерфейс әдістері мен қасиеттерінің орындалуы болмауы мүмкін, олар абстрактілі әдістерге және дерексіз сыныптардың қасиеттеріне жақын болады. Бұл жағдайда интерфейс белгілі бір қозғалысты білдіретін Move әдісін анықтайды. Оның орындалуы жоқ, ешқандай параметр болмайды және ештеңе қайтармайды.

Дәл сол нәрсе Name қасиетіне қатысты. Бір қарағанда, бұл автоматты мүлік сияқты. Бірақ іс жүзінде, бұл интерфейстегі жылжымайтын мүлікті анықтамайтын, іске асырылмайтын сипаттама.

Интерфейсті декларациялаудың тағы бір сәті: егер оның мүшелері - әдістер мен қасиеттерде қол жетімділік модификаторлары болмаса, бірақ іс жүзінде әдепкі қол жетімділік жалпыға қол жетімді, өйткені интерфейстің мақсаты сыныптың оны іске асыру үшін функционалдығын анықтау болып табылады. Бұл тұрақты және тұрақты айнымалыларға да қатысты, олар класстарда және құрылымдарда әдепкі бойынша жеке модификаторы болады. Интерфейстерде олар әдепкі бойынша жалпы модификаторға ие. Мысалы, minSpeed тұрақты және жылжымалы интерфейстің максималды жылдамдығы туралы айтуға болады:

```
static void Main(string[] args)
{
    Console.WriteLine(IMovable.maxSpeed);
    Console.WriteLine(IMovable.minSpeed);
}
```

2.10 Сурет – мысалы Imovable интерфейсінің айнымалылары

Сонымен бірге, C # 8.0 бастап, интерфейс компоненттері үшін қол жетімділік модификаторларын нақты көрсете аламыз:

```
interface IMovable
{
    public const int minSpeed = 0;     // минимальная скорость
    private static int maxSpeed = 60;   // максимальная скорость
    public void Move();
    protected internal string Name { get; set; }    // название
    public delegate void MoveHandler(string message);  // определение делегата для события
    public event MoveHandler MoveEvent;    // событие движения
}
```

2.11 Сурет – Жетімділік модификаторлар мысалы

C # 8.0 бастап интерфейстер әдепкі әдістер мен қасиеттерді іске асыруды қолдайды. Бұл қарапайым әдістер мен қасиеттерді қарапайым класстар мен құрылымдардағы сияқты жүзеге асырылатын интерфейстерде анықтай алатынымызды білдіреді. Мысалы, Жылжыту әдісінің әдепкі орындалуын анықтаңыз:

```
interface IMovable
{
    // реализация метода по умолчанию
    void Move()
    {
        Console.WriteLine("Walking");
    }
}
```

2.12 Сурет - әдепкі әдістер мен қасиеттерді іске асыру

Интерфейстерде әдепкі сипаттарды енгізу біршама күрделірек, өйткені интерфейстерде статикалық емес айнымалыларды анықтай алмаймыз, сондықтан интерфейс қасиеттеріндегі объектінің күйін басқара алмаймыз. Сонымен қатар, біз қасиеттерге арналған әдепкі іске асыруды анықтай аламыз:

```
interface IMovable
{
    void Move()
    {
        Console.WriteLine("Walking");
    }
    // реализация свойства по умолчанию
    // свойство только для чтения
    int MaxSpeed { get { return 0; } }
}
```

2.13 Сурет – Әдепкі әдістер мен қасиеттерді іске асыру

Айта кету керек, егер интерфейстің жеке әдістері мен қасиеттері болса (яғни, жеке модификатормен), онда оларда әдепкі орындалуы болуы керек. Бұл кез-келген статикалық әдістер мен қасиеттерге қатысты (жеке емес):

```
    public const int minSpeed = 0;     // минимальная скорость
    private static int maxSpeed = 60;   // максимальная скорость
        // находим время, за которое надо пройти расстояние distance со скоростью speed
    static double GetTime(double distance, double speed) => distance / speed;
    static int MaxSpeed
    {
        get { return maxSpeed; }
        set
        {
            if (value > 0) maxSpeed = value;
        }
    }
}
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine(IMovable.MaxSpeed);
        IMovable.MaxSpeed = 65;
        Console.WriteLine(IMovable.MaxSpeed);
        double time = IMovable.GetTime(100, 10);
```

2.13 Сурет – Интерфейстің жеке әдістерінің әдепкі орындалуы

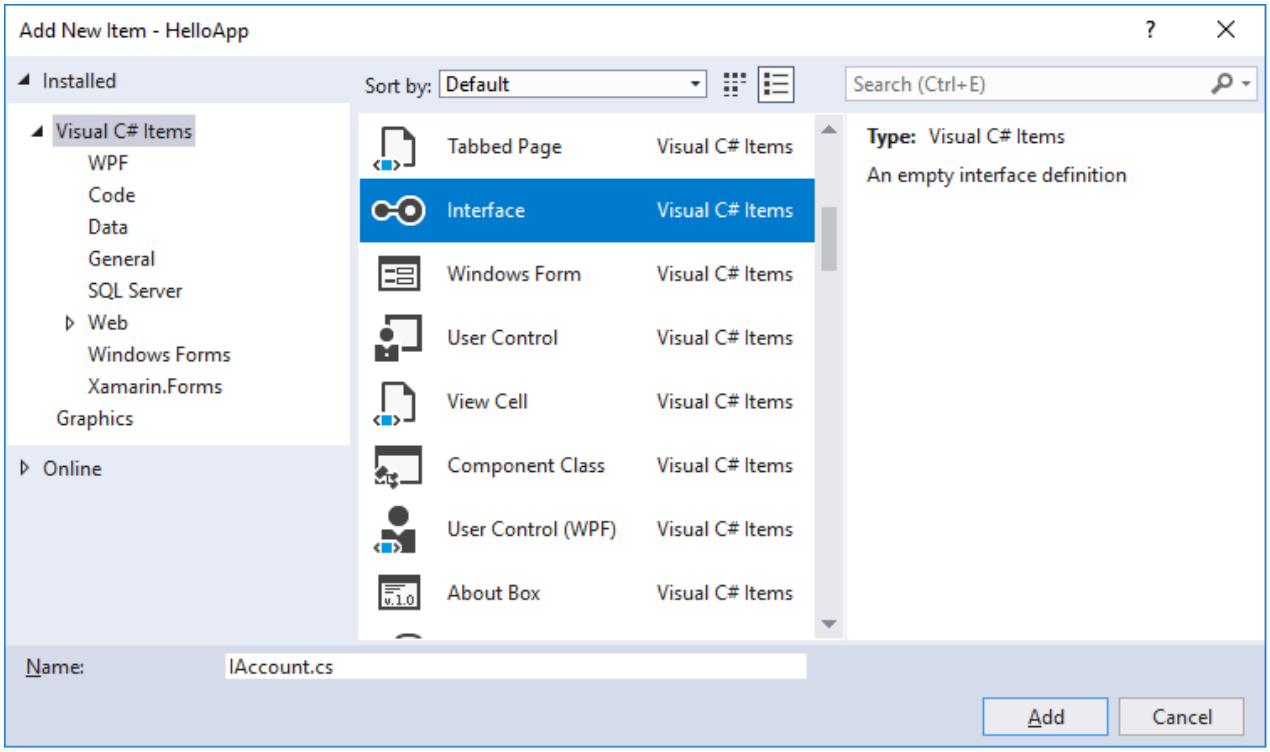
### 2.2.2 Интерфейске қол жеткізу модификаторлары

Класстар сияқты, әдепкі бойынша интерфейстердің ішкі қол жетімділік деңгейі бар, яғни мұндай интерфейс тек ағымдағы жоба аясында қол жетімді. Жалпы модификацияны қолдана отырып, біз интерфейсті жалпыға қол жетімді ете аламыз:

```
public interface IMovable
{
    void Move();
}
```

2.14 Сурет – Public модификаторының еңгізу мысалы

Айта кету керек, Visual Studio-да жаңа файлға жаңа интерфейс қосуға арналған арнайы компонент бар. Жобаға интерфейс қосу үшін, сіз жобаны тінтуірдің оң жақ түймешігімен нұқып, пайда болатын контекстік мәзірден Add-> New item ... таңдаңыз және жаңа компонент қосу тілқатысу терезесіндегі Интерфейсті таңдаңыз:




2.15 Сурет – Жаңа файлға жаңа интерфейс қосу

Класста интерфейсті енгізу үшін осы интерфейсте сипатталған әдістердің денелері (яғни нақты енгізулер) қамтамасыз етілуі керек. Әр сыныпқа интерфейсті іске асырудың егжей-тегжейін анықтауға толық еркіндік беріледі. Сондықтан бірдей интерфейсті екі сыныпта басқаша орындауға болады. Соған қарамастан, олардың әрқайсысы осы интерфейс үшін бірдей әдістер жиынтығын қолдауы керек. Мұндай интерфейс белгілі кодта осы екі класстың кез-келген объектісін пайдалануға болады, өйткені барлық осы объектілердің интерфейсі бірдей болып қалады. C # тіліндегі интерфейстердің арқасында полиморфизмнің негізгі қағидасы толығымен іске асырылуы мүмкін: бір интерфейс - көптеген әдістер.

```
interface имя{
    возвращаемый_тип имя_метода_1 (список_параметров);
    возвращаемый_тип имя_метода_2 (список_параметров);
    // ...
    возвращаемый_тип имя_метода_N (список_параметров);
}
```

2.16 Сурет – Жаңа бөлімдерді еңгізу

мұнда атау - интерфейстің нақты атауы. Интерфейс әдістерінің декларациясы олардың return_type және қолтаңбаларын ғана қолданады. Олар негізінен дерексіз әдістер. Жоғарыда айтылғандай, интерфейсте жүзеге асыру мүмкін емес. Сондықтан интерфейстің барлық әдістері осы интерфейсті қамтитын әр сыныпта орындалуы керек. Интерфейстің өзінде әдістер ашық болып саналады, сондықтан оларға қол жетімділікті нақты көрсету қажет емес.

Әдістерден басқа, сіз интерфейстердегі қасиеттерді, индекстегіштерді және оқиғаларды көрсете аласыз. Интерфейстерде деректер мүшелері болмайды. Олар сонымен қатар конструкторларды, деструкторларды немесе оператор әдістерін анықтай алмайды. Сонымен қатар, интерфейстің ешқандай мүшесі статикалық деп жариялана алмайды.

Интерфейс анықталғаннан кейін оны бір немесе бірнеше кластарда қолдануға болады. Интерфейсті іске қосу үшін оның атауын негізгі сыныпқа ұқсас класс атауынан кейін көрсету жеткілікті.

```
class имя_класса : имя_интерфейса {
    // тело класса
}
```

2.17 – Сурет класста интерфейсті імке асырудың жалпы формасы

мұнда interface_name - іске асырылатын интерфейстің нақты атауы. Егер интерфейс сыныпта енгізілсе, онда бұл толық орындалуы керек. Атап айтқанда, интерфейсті таңдамалы және тек бөліктерге ғана енгізу мүмкін емес.

Сыныпта бірнеше интерфейстерді қолдануға рұқсат етіледі. Бұл жағдайда сыныпта енгізілген барлық интерфейстер үтірмен бөлінген тізіммен көрсетіледі. Сыныпта сіз базалық классты мұра етіп, сонымен бірге бір немесе бірнеше интерфейстерді жүзеге асыра аласыз. Бұл жағдайда базалық класстың атауы үтірлермен бөлінген интерфейстер тізімінің алдында көрсетілуі керек.

Интерфейсті қолданатын әдістер жалпыға ортақ деп жариялануы керек. Шындығында, интерфейстің өзінде бұл әдістер ашық деп түсініледі, сондықтан оларды жүзеге асыру да ашық болуы керек. Сонымен қатар, орындалған әдістің қайтару түрі мен қолтаңбасы интерфейс анықтамасында көрсетілген қайтару түрі мен қолтаңбаға сәйкес келуі керек.

## 2.3 Деректер модельдері: концептуалды, логикалық және физикалық

Деректер моделі, яғни пәндік саланың тұжырымдамалық сипаттамасы – деректер базасын жобалаудың ең абстрактілі деңгейі. Пәндік саланы анықтаудың негізгі үлгісі мәні/байланыс (ER-әдіс) типінің моделі болып

табылады. Бұл модель пән аймағының семантикалық сипаттамасын және деректердің негізгі құрылымдарын анықтау үшін ақпаратты ұсынуды анықтайды. Модельді сипаттау элементтері мәні, атрибуттары және қатынастары болып табылады.

Мәні-бұл ақпаратты өңделетін жүйеде сақтау керек нәрсе. Көптеген мәндер нақты әлемнің объектілерін немесе оқиғаларын модельдейді, мысалы клиенттер, тауарлар және т. б. болуы мүмкін. Мәндер де абстрактілі ұғымдарды модельдей алады. Ең жарқын мысал – мәндер арасындағы қатынастарды моделдейтін мән: мысалы, белгілі бір сауда агенті белгілі бір клиентке жауап береді.

Бұл жүйе әр нысанның параметрлері сақталады. Бұл параметрлер нысан атрибуттары деп аталады. Өңделетін модельге қосылатын атрибуттарды анықтау-семантикалық процесс. Бұл міндетті шеше отырып, сақталатын деректерді шын мәнінде білдіретінін және олар қалай пайдаланылатынын негізге алу қажет.

Әрбір мәннің атрибуттарынан басқа, деректер моделі мәндер арасындағы байланысты анықтауы тиіс. Байланыстың концептуалды деңгейінде мәндер арасындағы қарапайым ассоциациялар болып табылады.

Деректердің логикалық моделі болашақ деректер базасында тіркеуге жататын фактілер мен объектілерді сипаттайды. Мұндай модельдің негізгі компоненттері олардың мәні, атрибуттары және арасындағы байланыс болып табылады. Әдетте, болашақ деректер базасындағы мәннің физикалық аналогы - кесте, ал атрибуттың физикалық аналогы-осы кестенің өрісі.

Логикалық тұрғыдан мәні бір типті объектілер немесе осы мәннің даналары деп аталатын фактілер жиынтығы болып табылады.

Дананың физикалық аналогы әдетте деректер қорының кестесіндегі жазба болып табылады. Және жазу-кестеде реляционды ДҚБЖ даналары мәнін бірегей болуы керек, яғни толық жинағы мәндерін олардың атрибуттарын тиіс қайталануы. Кестедегі өрістер сияқты атрибуттар кілтті және кілтті емес болуы мүмкін.

Деректерді физикалық жобалау логикалық модель негізінде жүзеге асырылады. Бұл процестің нәтижесі деректер базасында барлық қажетті объектілерді генерациялау үшін қажетті толық ақпаратты қамтитын физикалық модель болып табылады. Жүйелік каталогты қолдайтын ДҚБЖ үшін физикалық модель оның мазмұнына сәйкес келеді.

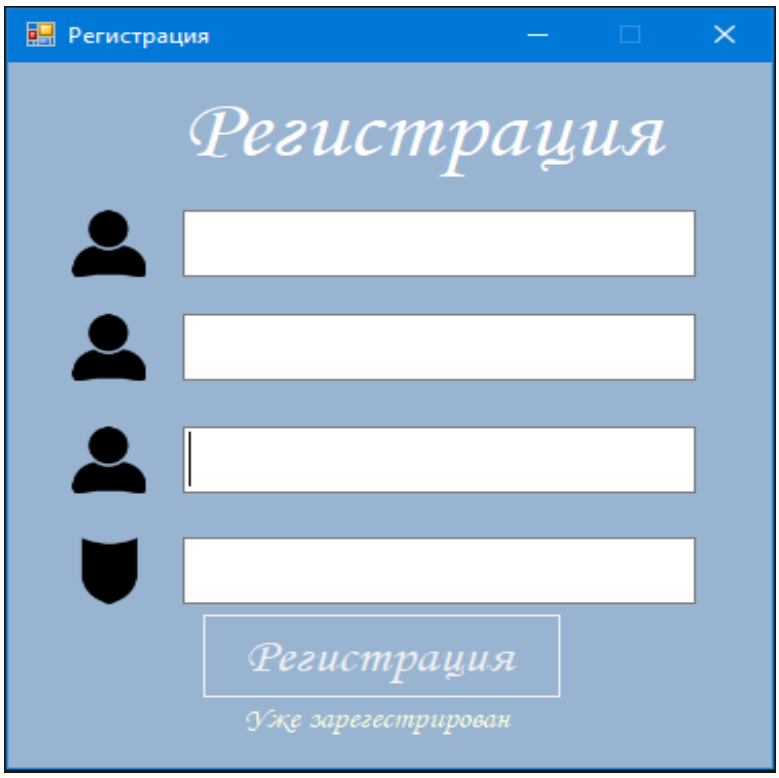
Физикалық жобалау процесінде барлық өрістер үшін кестелердің атаулары мен деректер типтерін анықтау керек. Егер қажет болса, осы кезеңде көріністер (егер ондайлар жасалатын болса) сипатталады және сақталатын процедуралардың коды жасалуы мүмкін. Мысалы, қандай SQL-ұсыныстардың көмегімен индекстер құрылады, қандай объектілер — триггерлер немесе серверлік шектеулердің көмегімен-сілтемелік тұтастық іске асырылады, баламалы кілттер мен Т. Б. үшін индекстер құрылады ме.

## 3 Жасалған жүйенің бағдарламасын іске асыру

Бұл бағдарламаның жасалу барысында Visual studio қосымша бағдарламасы қолданылғын. Бұл бағдарламада үш терезе бар: тіркеу терезесі, авторизация терезесі және уақыт кестені оқу панелі.

```
private bool IsLogin
{
    get
    {
        bool been = false;
        string loginUser = textBoxLogin.Text;
        string passwordUser = textBoxPassword.Text;

        DatabaseManager _databaseManager = new DatabaseManager();
        DataTable _dataTable = new DataTable();
        MySqlDataAdapter _mySqlDataAdapter = new MySqlDataAdapter();
        MySqlCommand _mySqlCommand = new MySqlCommand("SELECT * FROM `users` WHERE `login` = @UserLogin AND `password` = @UserPassword", _databaseManager.GetConnection);

        //меняем заглушки на переменные
        _mySqlCommand.Parameters.Add("@UserLogin", MySqlDbType.VarChar).Value = loginUser;
        _mySqlCommand.Parameters.Add("@UserPassword", MySqlDbType.VarChar).Value = passwordUser;

        _mySqlDataAdapter.SelectCommand = _mySqlCommand;//выбираем команду
        _mySqlDataAdapter.Fill(_dataTable);//заполянем данные в табличку ***

        if (_dataTable.Rows.Count > 0)
        {
            been = true;
            MessageBox.Show("Такой логин уже есть!\nПопробуйте другой логин!", "Внимание!");
        }
        else
            been = false;

        return been;
    }
}
```

3.1 сурет - Тіркелу терезесінің C# тіліндегі коды



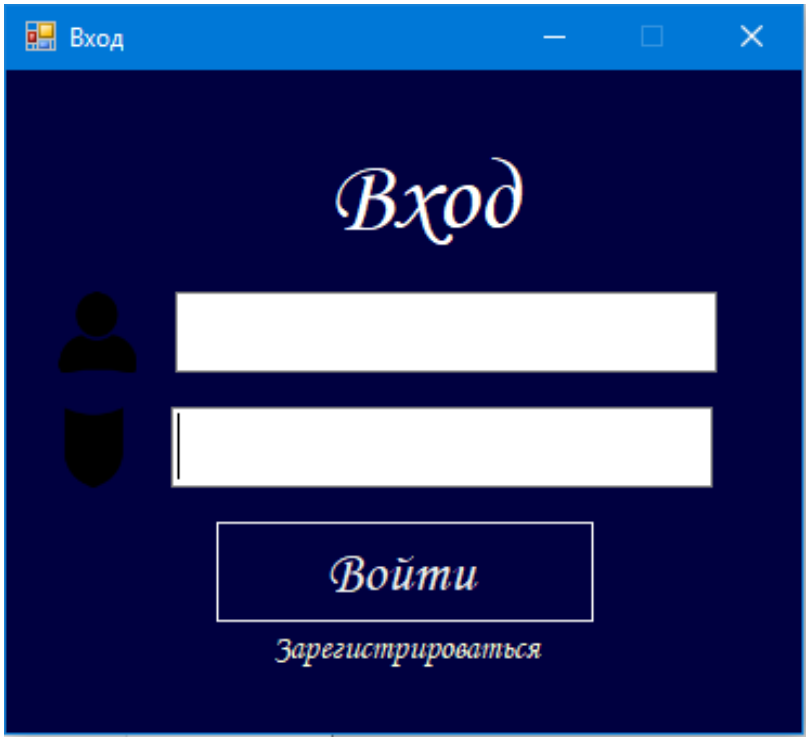

3.2 сурет – Тіркелу терезесі

Бұл бағдарлама бізбен кездесетін алғашқы терезе. Көріп отырғаныңыздай, сіз осы терезеде тіркеле аласыз, бірақ егер сіз бұрын тіркелген болсаңыз, келесі терезені - авторизация терезесін ашуға болады.

```
private void ButtonEnter_Click(object sender, EventArgs e)
{
    string loginUser = textBoxLogin.Text;
    string passwordUser = textBoxPassword.Text;

    DatabaseManager _databaseManager = new DatabaseManager();
    DataTable _dataTable = new DataTable();
    MySqlDataAdapter _mySqlDataAdapter = new MySqlDataAdapter();
    MySqlCommand _mySqlCommand = new MySqlCommand("SELECT * FROM `users` WHERE `login` = @UserLogin AND `password` = @UserPassword", _databaseManager.GetConnection);

    try
    {
        //меняем заглушки на переменные
        _mySqlCommand.Parameters.Add("@UserLogin", MySqlDbType.VarChar).Value = loginUser;
        _mySqlCommand.Parameters.Add("@UserPassword", MySqlDbType.VarChar).Value = passwordUser;

        _databaseManager.OpenConnection();//открываем соеденения

        _mySqlDataAdapter.SelectCommand = _mySqlCommand;//выбираем команду
        _mySqlDataAdapter.Fill(_dataTable);//заполянем данные в табличку

        if (_dataTable.Rows.Count > 0)
        {
            if(loginUser != "Admin")
            {
                //елси уже зарегестрирован
                WorkerForm form = new WorkerForm();
                this.Hide();
                form.Show();
            }
            else
            {
                AdminForm form = new AdminForm();
                this.Hide();
                form.Show();
            }

            //запомним кто вошел
            User user = new User(loginUser);
        }
        else
        {
```

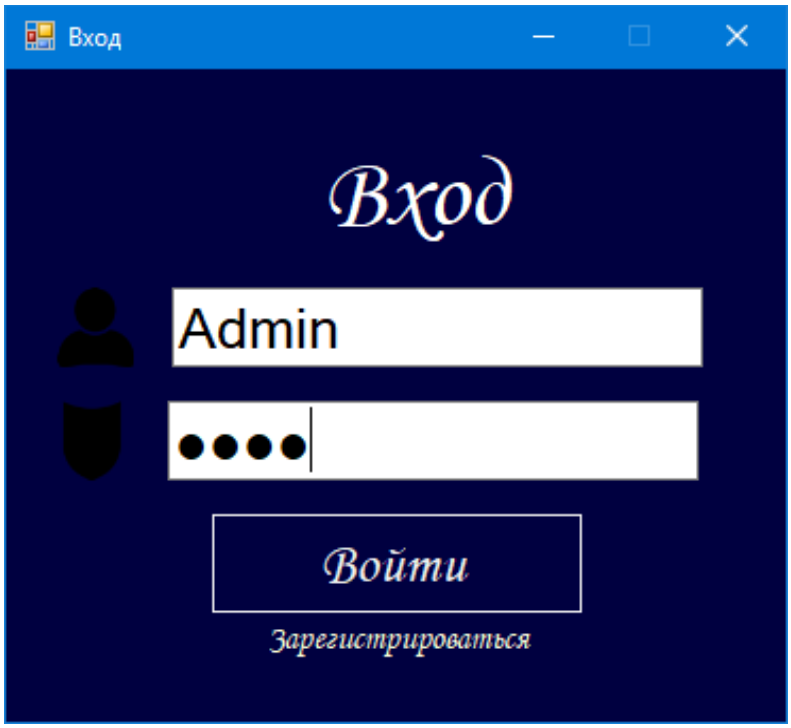
```
            else
            {
                //проверим, если такого логина нет
                if (IsLogin)
                {
                    MessageBox.Show("Пароль введен не верно!", "Внимание!");
                }
                else
                {
                    if (MessageBox.Show("Вы у нас впервые!\nНеобходимо зарегистрироваться!\nЗарегистрироваться сейчас?",
                        "Внимание!", MessageBoxButtons.YesNo) == DialogResult.Yes)
                    {
                        Registration form = new Registration();
                        this.Hide();
                        form.Show();
                    }
                }

            }
        }
        catch
        {
            MessageBox.Show("Ошибка работы с базой данных!", "Ошибка");
        }
        finally
        {
            _databaseManager.CloseConnection();//закрываем соеденение
        }
    }

    ссылка: 1
    private void labelRegistration_Click(object sender, EventArgs e)
    {
        Registration form = new Registration();
        form.Show();
        this.Hide();
    }
}
```

3.3 сурет – Авторизация терезесінің C#тіліндегі коды




3.4 сурет – Авторизация терезесі

Сіз сәтті  логинді тіркегеннен кейін бағдарлама сізді тіркеу терезесіне жібереді, осылайша сіз өзіңіз жазған логин мен парольді енгізіп программада тіркелесіп программа сіздің келген уақытыңызды деректер қорындағы арнайы кестеге еңгізеді.



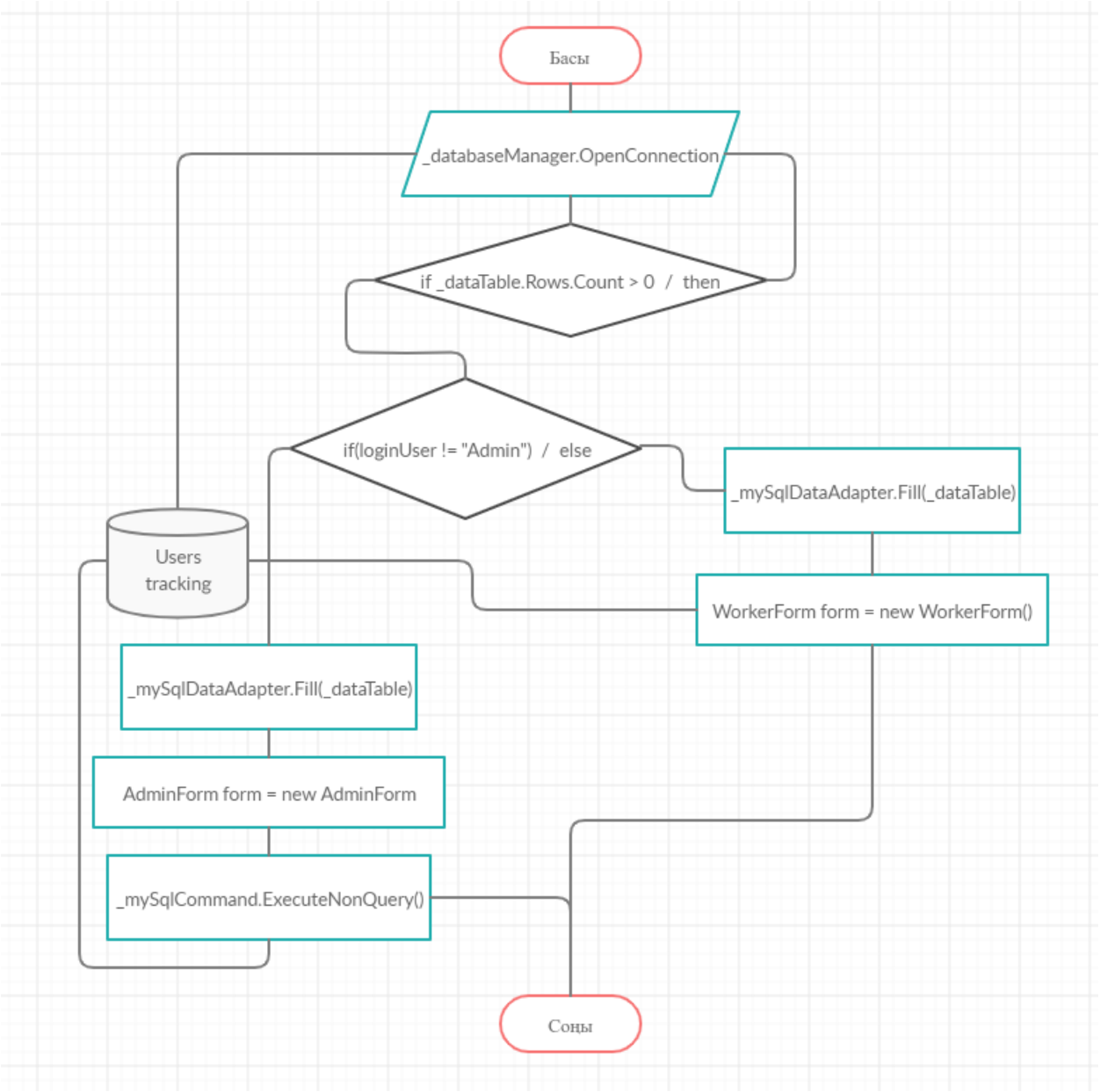

3.5 сурет – Авторизация терезесі (мысал)

Егер сіз менеджердің есептік жазбасынан кірсеңіз, онда сіз барлық қызметкерлермен бірге кестені және олардың жұмысқа келген уақыты мен кететін уақытын көресіз. Келесі 3.6 суретте көріп отырғаныңыздай арнайы менеджердің басқарушы кесеті ашылады.

| | Логин | Прибыл | Убыл |
|---|---|---|---|
| ▶ | user1 | 14.05.2020 5:25:52 | work |
| | user1 | 14.05.2020 5:25:52 | 14.05.2020 5:26:05 |
| | admin | 14.05.2020 5:26:14 | work |
| | Karin.A | 14.05.2020 6:34:59 | work |
| | Karin.A | 14.05.2020 6:34:59 | 14.05.2020 6:35:04 |
| | Shapenov.A | 14.05.2020 6:35:38 | work |
| | Shapenov.A | 14.05.2020 6:35:38 | 14.05.2020 6:35:42 |
| | Tolebay.K | 14.05.2020 6:36:09 | work |
| | Tolebay.K | 14.05.2020 6:36:09 | 14.05.2020 6:36:12 |
| | Eleusynov.R | 14.05.2020 6:36:46 | work |
| | Eleusynov.R | 14.05.2020 6:36:46 | 14.05.2020 6:37:00 |

3.6 сурет – Басқарушы кестесі

Көріп отырғаныңыздай, бағдарламаға менеджердің есеп-шотынан кіргенде, бүгін жұмысқа келген қызметкерлер мен олар келген уақыт кестесі бар терезе ашылады.




3.7 Сурет – Жасалған бағдарламаның жұмыс алгоритмі

Үстіде көрсетілген алгоритм бағдарламаның толықтай жұмысын түсіндіреді, яғни бағдарлама қосылған кезде, біз бағдарламаға деректерді енгіземіз (тіркеу терезесінде логин паролін енгізу), бағдарлама біздің деректерді қабылдап, деректер базасымен байланыс орнатады. Деректер базасымен байланыс орнатқаннан кейін, бағдарлама біздің деректерімізді өзінде бар деректермен салыстырады, бағдарламада сәтті авторизациядан кейін біздің деректерімізді жазып алып (қызметкерлердің тіркеу уақыты) жаңа терезе ашылады. Жаңа терезеде жұмысшылар үшін бағдарлама туралы мәліметтермен және қарапайым пайдаланушылар мәзірімен ашылады. Егер тіркеу кезінде бағдарлама менеджердің мәліметтерін көретін болса, онда арнайы қызметкерлердің жеке терезесі ашылады, ол барлық қызметкерлердің деректерімен кесте және әр қызметкердің келу уақытын көрсетеді. Сондай-ақ,

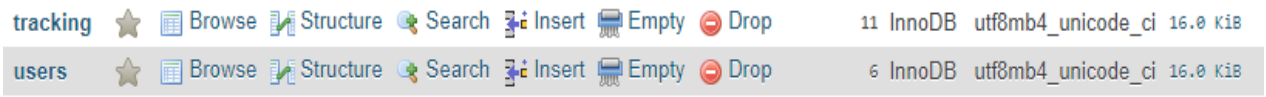
осы терезеде менеджер қызметкер туралы жаңа деректерді еңгізіп, оларды өзгерту және жою үшін қолдана алады.

### 3.1 Деректер базасын әзірлеу

Осы жоба үшін әзірленген деректер базасы тек екі кестеден тұрады, олар:

Tracking - Тіркелеген жұмысшылардың кірген шыққан уақытын жазып алатын кесте.

Users – бұл кестеде әр тілкелеген адамның логин паролі тегі аты енгізілген.




3.8 сурет – ДБ құрылымы

```
namespace _108_Time_tracking_for_workers
{
    Ссылок: 16
    class DatabaseManager
    {
        //создаем поле данных соеденения с MySql
        MySqlConnection connection = new MySqlConnection("server=localhost;" +
            "port=3307;" +
            "username=root;" +
            "password=root;" +
            "database=(108)tracking_workers");

        //метод, который открывает соеденение
        Ссылок: 5
        public void OpenConnection()
        {
            //проверяем состояние подключения
            if (connection.State == System.Data.ConnectionState.Closed)
                connection.Open();
        }

        //закрываем соеденение
        Ссылок: 2
        public void CloseConnection()
        {
            //проверяем, есть ли соеденение
            if (connection.State == System.Data.ConnectionState.Open)
                connection.Close();
        }

        //возращаем значения объекта соеденения
        Ссылок: 8
        public MySqlConnection GetConnection { get { return connection; } }

    }
}
```

3.9 сурет – ДБ мен қосылыс

### 3.2 Пайдаланушы интерфейсін әзірлеу

Осы бағдарламаны пайдалану үшін пайдаланушы кез келген компьютерді пайдалана алады. Қосымша бағдарламалық жасақтаманы орнату қажет емес. Бағдарламаны іске қосу кезінде алдымен авторизацилану керек

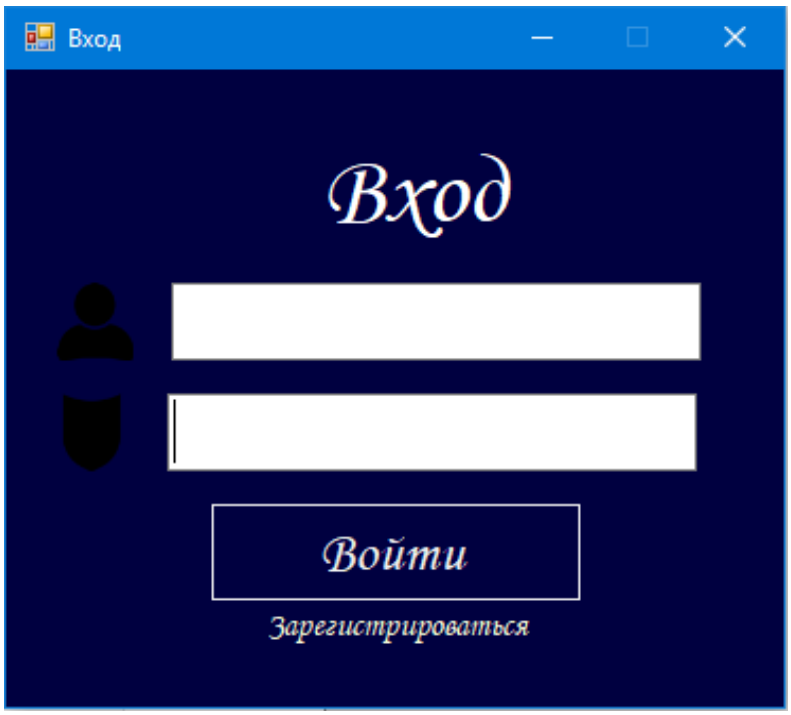
```
Ссылок: 11
public partial class Authorization : Form
{
    ссылка: 1
    public bool IsLogin
    {
        get
        {
            bool been = false;
            string loginUser = textBoxLogin.Text;
            string passwordUser = textBoxPassword.Text;

            DatabaseManager _databaseManager = new DatabaseManager();
            DataTable _dataTable = new DataTable();
            MySqlDataAdapter _mySqlDataAdapter = new MySqlDataAdapter();
            MySqlCommand _mySqlCommand = new MySqlCommand("SELECT * FROM `users` WHERE `login` = @UserLogin", _databaseManager.GetConnection);

            //меняем заглушки на переменные
            _mySqlCommand.Parameters.Add("@UserLogin", MySqlDbType.VarChar).Value = loginUser;

            _mySqlDataAdapter.SelectCommand = _mySqlCommand;//выбираем команду
            _mySqlDataAdapter.Fill(_dataTable);//заполянем данные в табличку ***

            if (_dataTable.Rows.Count > 0)
            {
                been = true;
            }
            else
                been = false;

            return been;
        }
    }

    Ссылок: 4
    public Authorization()
    {
        InitializeComponent();
    }

    ссылка: 1
    private void ButtonEnter_Click(object sender, EventArgs e)
```

```
private void ButtonEnter_Click(object sender, EventArgs e)
{
    string loginUser = textBoxLogin.Text;
    string passwordUser = textBoxPassword.Text;

    DatabaseManager _databaseManager = new DatabaseManager();
    DataTable _dataTable = new DataTable();
    MySqlDataAdapter _mySqlDataAdapter = new MySqlDataAdapter();
    MySqlCommand _mySqlCommand = new MySqlCommand("SELECT * FROM `users` WHERE `login` = @UserLogin AND `password` = @UserPassword", _databaseManager.GetConnection);

    try
    {
        //меняем заглушки на переменные
        _mySqlCommand.Parameters.Add("@UserLogin", MySqlDbType.VarChar).Value = loginUser;
        _mySqlCommand.Parameters.Add("@UserPassword", MySqlDbType.VarChar).Value = passwordUser;

        _databaseManager.OpenConnection();//открываем соеденения

        _mySqlDataAdapter.SelectCommand = _mySqlCommand;//выбираем команду
        _mySqlDataAdapter.Fill(_dataTable);//заполянем данные в табличку

        if (_dataTable.Rows.Count > 0)
        {
            if(loginUser != "Admin")
            {
                //елси уже зарегестрирован
                WorkerForm form = new WorkerForm();
                this.Hide();
                form.Show();
            }
            else
            {
                AdminForm form = new AdminForm();
                this.Hide();
                form.Show();
            }

            //запомним кто вошел
            User user = new User(loginUser);
        }
        else
```

3.10 сурет – Авторизация терезесі толық код түрінде

3.10 суретің жалғасы

```
            else
            {
                //проверим, если такого логина нет
                if (IsLogin)
                {
                    MessageBox.Show("Пароль введен не верно!", "Внимание!");
                }
                else
                {
                    if (MessageBox.Show("Вы у нас впервые!\nНеобходимо зарегистрироваться!\nЗарегистрироваться сейчас?",
                        "Внимание!", MessageBoxButtons.YesNo) == DialogResult.Yes)
                    {
                        Registration form = new Registration();
                        this.Hide();
                        form.Show();
                    }
                }

            }
        }
        catch
        {
            MessageBox.Show("Ошибка работы с базой данных!", "Ошибка");
        }
        finally
        {
            _databaseManager.CloseConnection();//закрываем соеденение
        }
    }

    ссылка: 1
    private void labelRegistration_Click(object sender, EventArgs e)
    {
        Registration form = new Registration();
        form.Show();
        this.Hide();
    }
```




3.11 сурет – Авторизация терезесі

Бұл терезеде біз бағдарламада авторизациялауға арналған сызықтарды, сондай-ақ егер сізге әлі рұқсат берілмеген болса, тіркеу терезесіне сілтемені көре аламыз.

```
Ссылок: 5
public Registration()
{
    InitializeComponent();
}

//проверка логина
ссылка: 1
private bool IsLogin
{
    get
    {
        bool been = false;
        string loginUser = textBoxLogin.Text;
        string passwordUser = textBoxPassword.Text;

        DatabaseManager _databaseManager = new DatabaseManager();
        DataTable _dataTable = new DataTable();
        MySqlDataAdapter _mySqlDataAdapter = new MySqlDataAdapter();
        MySqlCommand _mySqlCommand = new MySqlCommand("SELECT * FROM `users` WHERE `login` = @UserLogin AND `password` = @UserPassword", _databaseManager.GetConnection);/

        //меняем заглушки на переменные
        _mySqlCommand.Parameters.Add("@UserLogin", MySqlDbType.VarChar).Value = loginUser;
        _mySqlCommand.Parameters.Add("@UserPassword", MySqlDbType.VarChar).Value = passwordUser;

        _mySqlDataAdapter.SelectCommand = _mySqlCommand;//выбираем команду
        _mySqlDataAdapter.Fill(_dataTable);//заполянем данные в табличку ***

        if (_dataTable.Rows.Count > 0)
        {
            been = true;
            MessageBox.Show("Такой логин уже есть!\nПопробуйте другой логин!", "Внимание!");
        }
        else
            been = false;

        return been;
    }
}

//проверка пользователя
ссылка: 1
private bool IsUser
```

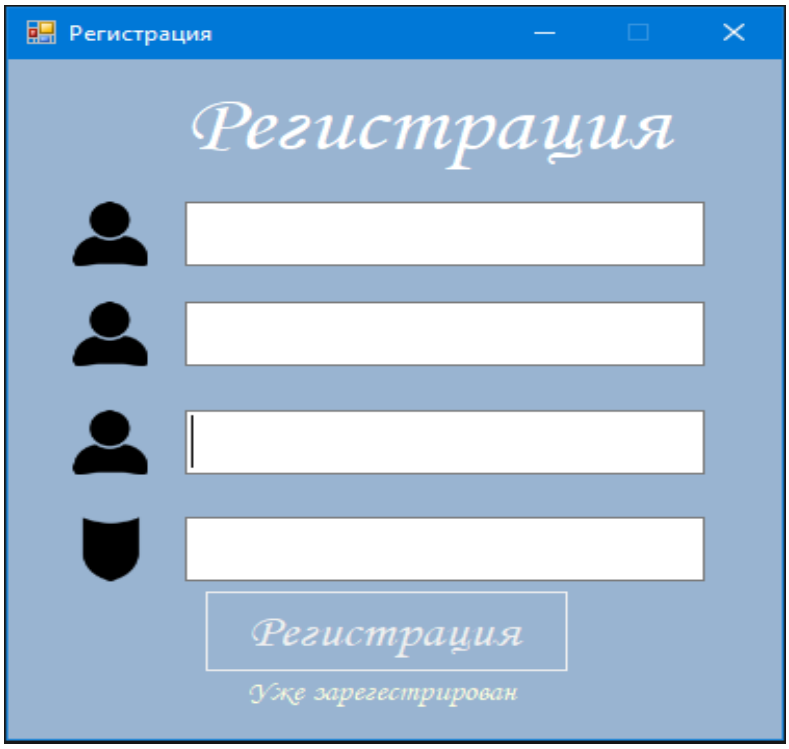
3.12 сурет – Тіркелу терезесінің толық коды

# 3.12 сурет жалғасы

```
private bool IsUser
{
    get
    {
        /*
         * Проверяем все данные, которые вводит пользователь на совпадение,
         * кроме пароля. Так как возможно пользователь уже есть, но он ввел
         * другой пароль.
         */

        bool been = false;

        string loginUser = textBoxLogin.Text;
        string nameUser = textBoxName.Text;
        string surnameUser = textBoxSurname.Text;

        DatabaseManager _databaseManager = new DatabaseManager();
        DataTable _dataTable = new DataTable();
        MySqlDataAdapter _mySqlDataAdapter = new MySqlDataAdapter();
        MySqlCommand _mySqlCommand = new MySqlCommand("SELECT * FROM `users` WHERE `login` = @UserLogin AND `name` = @UserName AND `surname` = @UserSurname", _databaseManager.GetConnection);

        //меняем заглушки на переменные
        _mySqlCommand.Parameters.Add("@UserLogin", MySqlDbType.VarChar).Value = loginUser;
        _mySqlCommand.Parameters.Add("@UserName", MySqlDbType.VarChar).Value = nameUser;
        _mySqlCommand.Parameters.Add("@UserSurname", MySqlDbType.VarChar).Value = surnameUser;

        _mySqlDataAdapter.SelectCommand = _mySqlCommand;//выбираем команду
        _mySqlDataAdapter.Fill(_dataTable);//заполянем данные в табличку

        //проверяем на совпадение
        if (_dataTable.Rows.Count > 0)
        {
            been = true;
            if (MessageBox.Show("Такой пользователь уже есть!\nПерейти на вкладку входа?", "Внимание!", MessageBoxButtons.YesNo) == DialogResult.Yes)
            {
                //переходим на форму входа
                Authorization form = new Authorization();
                this.Hide();
                form.Show();
            }
        }
        else
            //выполянем запрос
            _databaseManager.OpenConnection();//открываем соеденения

            if (_mySqlCommand.ExecuteNonQuery() == 1)
            {
                MessageBox.Show("Аккаует создан!", "Внимание!");

                WorkerForm form = new WorkerForm();
                this.Hide();
                form.Show();

                //запоним кто вошел
                User user = new User(textBoxLogin.Text);
            }
            else
                MessageBox.Show("Ошибка создания аккаунта!", "Ошибка!");

            _databaseManager.CloseConnection();//закрываем соеденение

    }
}
```

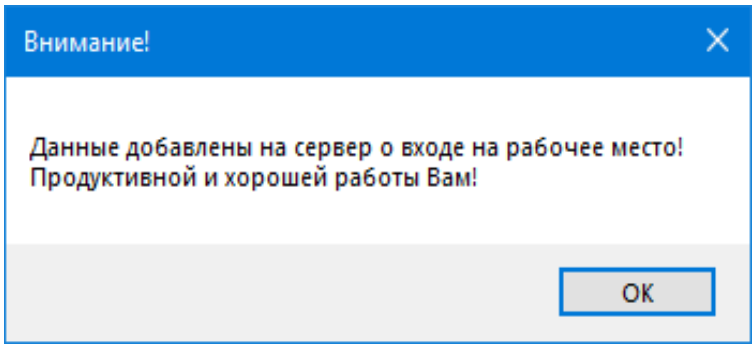
3.12 сурет жалғасы

```
            else
                been = false;


            return been;
        }
    }

    ссылка: 1
    private void ButtonRegistration_Click(object sender, EventArgs e)
    {
        //Проверка ввода данных
        if (textBoxName.Text == "" || textBoxLogin.Text == ""
           || textBoxPassword.Text == "" || textBoxSurname.Text == "")
        {
            MessageBox.Show("Не все поля введены!", "Внимание!");
            return;
        }

        //прверка на уникальность всех данных
        if (!IsUser)
        {
            //проверка на уникальность логина
            if (!IsLogin)
            {
                DatabaseManager _databaseManager = new DatabaseManager();
                MySqlCommand _mySqlCommand = new MySqlCommand("INSERT INTO `users`(`login`, `password`, `name`, `surname`) " +
                    "VALUES (@login,@password,@name,@surname)", _databaseManager.GetConnection);//формируем запрос

                //меняем заглужки на значения
                _mySqlCommand.Parameters.Add("@login", MySqlDbType.VarChar).Value = textBoxLogin.Text;
                _mySqlCommand.Parameters.Add("@password", MySqlDbType.VarChar).Value = textBoxPassword.Text;
                _mySqlCommand.Parameters.Add("@name", MySqlDbType.VarChar).Value = textBoxName.Text;
                _mySqlCommand.Parameters.Add("@surname", MySqlDbType.VarChar).Value = textBoxSurname.Text;

                //выполянем запрос
                _databaseManager.OpenConnection();//открываем соеденения

                if (_mySqlCommand.ExecuteNonQuery() == 1)
                {
                    MessageBox.Show("Аккаует создан!", "Внимание!");
```

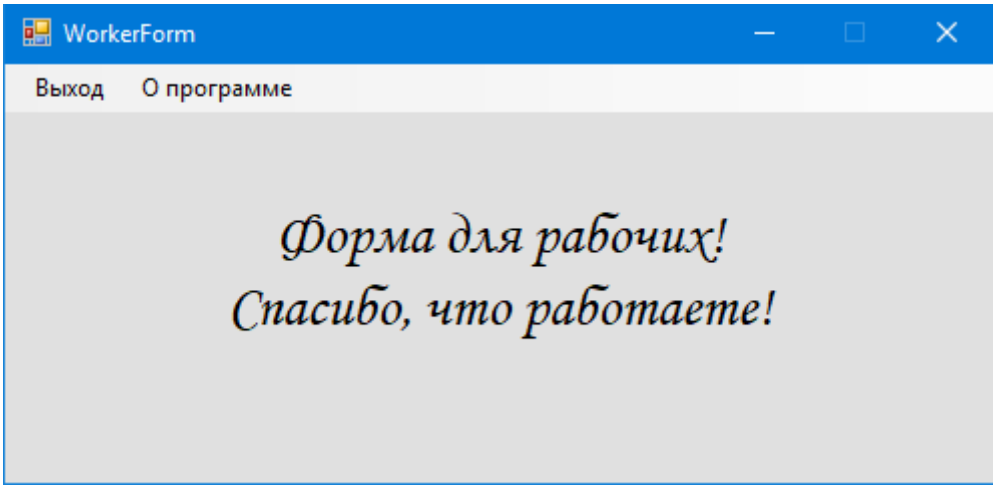



3.13 сурет – Тіркелу терезесі

Тіркеу терезесінде біз фамилияға арналған жолдарды, сонымен қатар бағдарламада авторизациялау үшін логин мен парольді көреміз. Тіркелу терезесін тек менеджер ғана ашып жұмысшыларды тіркей алады.




3.14 сурет – Бағдарламадан шығу хабарлама терезесі

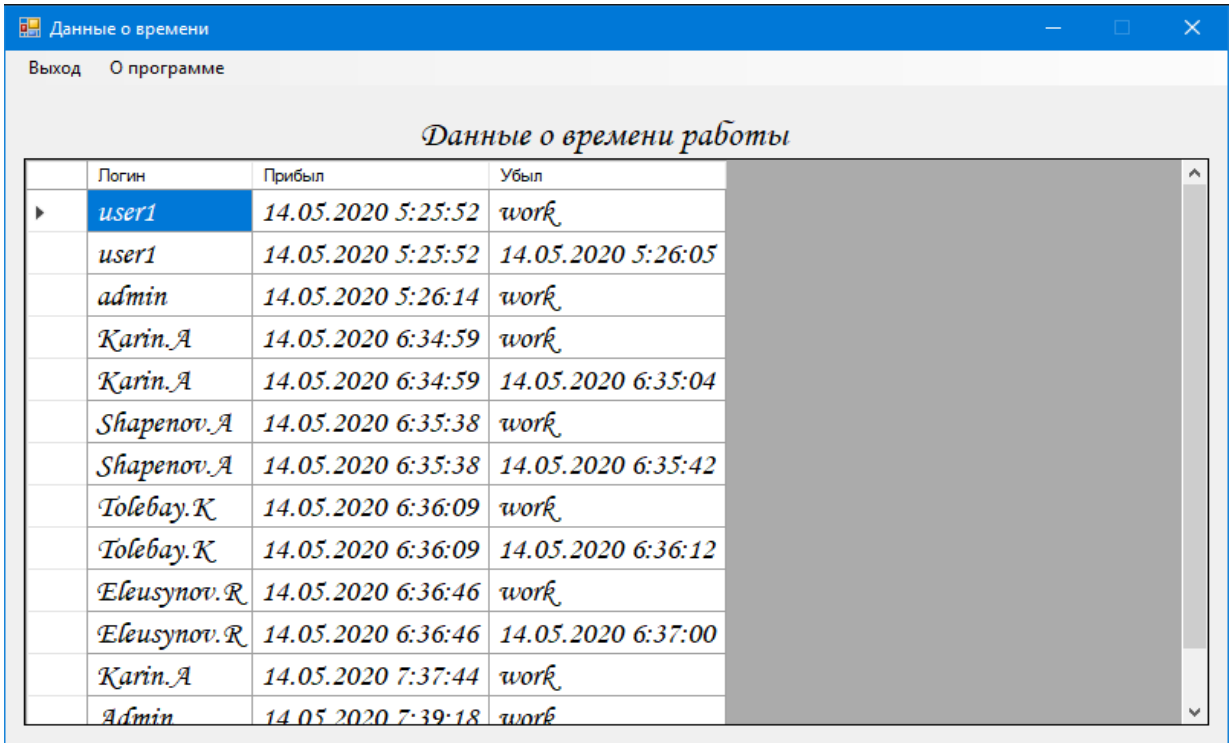



3.15 сурет – Сәтті авторизациядан кейінгі терезе

Көріп отырғанымыздай, сәтті тіркеуден өткеннен кейін сіз бағдарламаның өзіне кіре бастайсыз және сіздің жұмыс уақытыңызды тіркеу үшін авторизация уақыты жазылады.

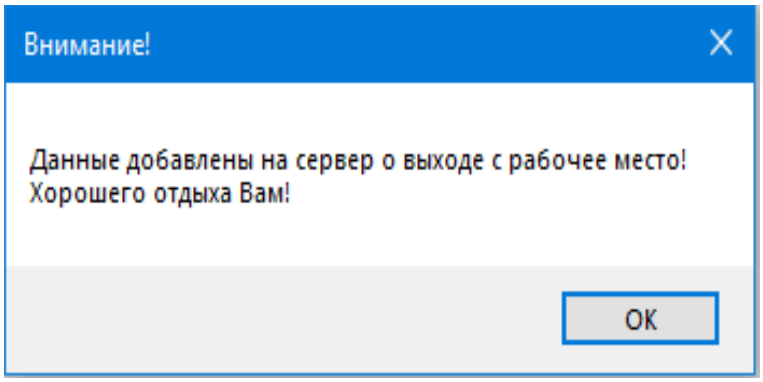
```
Ссылок: 4
public partial class AdminForm : Form
{
    ссылка: 1
    public AdminForm()
    {
        InitializeComponent();
    }

    ссылка: 1
    private void AdminForm_Shown(object sender, EventArgs e)
    {
        //открываем базу данных и считываем с нее данные
        DatabaseManager _manager = new DatabaseManager();
        MySqlCommand _command = new MySqlCommand("SELECT * FROM `tracking`", _manager.GetConnection);
        MySqlDataReader _reader;

        _manager.OpenConnection();
        _reader = _command.ExecuteReader();

        while (_reader.Read())
        {
            //заполняем данные
            dataGridView.Rows.Add(_reader["user"], _reader["arrival_time"], _reader["time_of_departure"]);
        }

        dataGridView.AutoSizeColumnsMode =
         DataGridViewAutoSizeColumnsMode.AllCells;
        dataGridView.AutoResizeColumns();

        dataGridView.AutoSizeRowsMode =
            DataGridViewAutoSizeRowsMode.AllCells;
        dataGridView.AutoResizeRows(
            DataGridViewAutoSizeRowsMode.AllCellsExceptHeaders);
    }

    ссылка: 1
    private void оПрограммеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        MessageBox.Show("Информация о программе!","Информация!");
    }
```

3.16 сурет – Администратор терезесінің толық коды



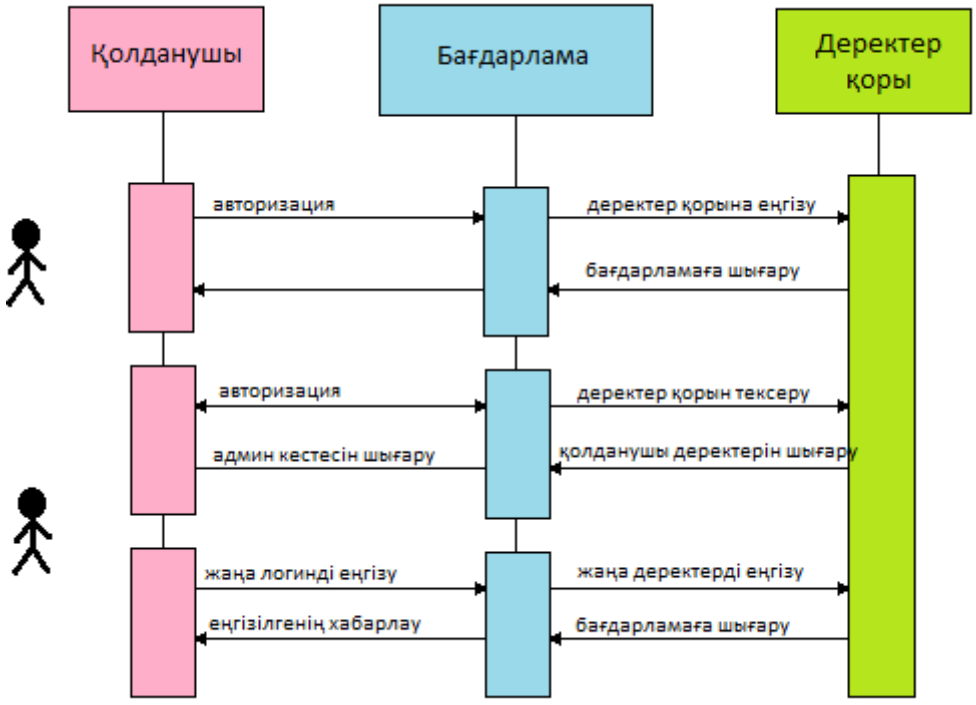

3.17 сурет – Администратор үшін терезе

Менеджердің есептік жазбасынан кіргенде, оның бөлімшесінің барлық қызметкерлерінің кестесі және одан әрі бағалау үшін жұмыс уақыты көрсетілген терезе қалай ашылатынын көреміз.




3.18 сурет – Бағдарламадан шыққаннан кейінгі хабарлау терезесі

Осы бағдарламаны жапқаннан кейін, ол автоматты түрде сіздің логиніңізді қалдырады және сіздің жұмыс күніңізді аяқтағаныңызды айтып, сіздің кіру уақытын жазып алады.




3.19 Сурет – Бағдарламамен ұолдану диаграммасы

## 4 Техникалық-экономикалық көрсеткіштер

### 4.1 Бағдарламаның өзіндік құнын есептеу

Бұл бөлімде «Кәсіпорын қызметкерлерінің жұмыс уақытын бақылаудың ақпараттық жүйесін құру» әзірленген бағдарламалық өнімнің техникалық-экономикалық көрсеткіштері есептелген. Бұл жоба жалақыны жұмыс уақытына қарап төйлейтін компаниялар үшін керек қосымша.

Жобаның экономикалық тиімділігін есептеу үшін оның өзіндік құнын есептеп, бағасын, маржасын есептеп, таза пайданы анықтау қажет.

Өндірістің өзіндік құны – тауар өндіруге жұмсалған шығын. Өнімнің өзіндік құнын келесі экономикалық элементтерді қосу арқылы аламыз:

- негізгі материалдар, сатып алынатын өнімдер (М);
- технологиялық мақсаттар үшін электр энергиясы (Эл.);
- еңбекақы төлеу (Z);
- әлеуметтік қажеттіліктерге аударымдар (Соц.);
- амортизациялық аударымдар (А.);
- басқа шығындар (Пр.).

«Материалдық шығындар» сәтінде материалдық ресурстардың құны көрсетілген, олардың жиынтығы олар сатып алынған бағаларға негізделеді және ешқандай қосымша төлемдерсіз, үстеме ақыларсыз, әртүрлі үшінші тараптардың қызметтерінің құны (кеден, қойма, көлік ұйымдары және т.б.) көрсетіледі. Сіз материалдық ресурстарға қайтарылатын қалдықтардың құнын алып тастай аласыз. Қайта өңделетін қалдық деп шикізат, материалдар, жартылай фабрикаттар және басқа да материалдық құндылықтардың қалдықтары оны өндіруге толық немесе ішінара пайдалануға болады. Мұндай қайта өңделетін қалдықтар әлеуетті пайдалану деңгейіне қарай бағаланады.

Технологиялық мақсаттар үшін электр энергиясына шығын бұл өнім өндіру процессінде кеткен электр энергия құны.

«Еңбекке ақы төлеу» элементі еңбек шығындарын, оның ішінде өндірістік персоналдың өндірістік шығындарын, өндіріс нәтижелерін, ынталандыру мен өтемақыны, сондай-ақ олардың негізгі қызметімен айналыспайтын жұмысшылардың еңбек ақыларын қамтиды.

«Әлеуметтік қажеттіліктерді алып тастау» элементтері әлеуметтік сақтандыру мекемелері, Зейнетақы қорлары және міндетті дәрі-дәрмек туралы заңнамаға сәйкес белгіленген барлық салықтарға шегеріледі. Олар жұмысшылардың жалақысына және еңбек ақыға (қызметтерге) қосылады («еңбек шығындары» үшін). Сақтандыру сыйлықақылары жалақының белгілі бір түрлері үшін төленбейді (мұндай төлемдер нормативтік құжатқа енгізілген).

Негізгі құралдардың амортизациясы – бұл құралдардың тозу дәрежесін көрсетеді.

Басқа шығындарға жиынтықтағы салықтар, төлемдер, сақтандыру қорларына салымдар, несие төлемдері, жол жүру, оқу, байланыс қызметтері, банктер, ақпараттық қызметтер және т.б. кіреді.

### 4.2 Бағдарламалық өнімді әзірлеудің еңбек салымы

Еңбек салымын анықтау үшін тапсырманы бірнеше кезеңге бөлу, осы кезеңде жүргізілетін жұмыстарды анықтау және жүктемені анықтау қажет, осылайша ПҚ әзірлеудің еңбек салымы алынады.

4.1 кесте – Жұмыс кезеңдері және олардың күрделілігі

| № | ПҚ-ны әзірлеу кезеңі | Кезеңде жасалатын жұмыс | Еңбек салымы | |
|---|---|---|---|---|
| | | | Адам х сағ. | Сағ. х күн |
| 1 | Талаптарды талдау | Пәндік саланы талдау, мақсаттар мен міндеттерді белгілеу. | 1 х 16 | 8 х 2 |
| 2 | Нарықты талдау | Компанияларды осындай шешімдерді пайдалану тәжірибесін талдау, функционалды егжей-тегжейлі қарау, артықшылықтар мен кемшіліктерді анықтау. Бағдарламалық өнімді әзірлеу қажеттілігін анықтау және бәсекелестік шешімдер аясында ерекшеліктерді анықтау. | 1 х 16 | 8 х 2 |
| 3 | Жобалау | Техникалық тапсырманы әзірлеу. Әзірлеу үшін қаражат жинағын анықтау. | 1 х 40 | 8 х 5 |
| 4 | Жүзеге асыру | Программалық қамтаманы әзірлеу | 1 х 120 | 8 х 15 |
| 5 | Тестілеу | Программаны қондырып тестілеу, ең жақсы нәтижелермен салыстыру | 1 х 24 | 8 х 3 |
| 6 | Енгізу және қолдау | Бағдарламалық өнімді пайдалануға енгізу, персоналды өніммен таныстыру және оны сүйемелдеу. | 1 х 16 | 8 х 2 |

### 4.3 Технологиялық мақсаттар үшін электр энергиясына кеткен шығынды есептеу

Технологиялық мақсаттар үшін электр энергиясының шығыны (Эл.) мына формула бойынша анықталады:

$$Эл. = К*Т*Ц, \quad (4.1)$$

К – сағатына компьютер қуат тұтынуы, кВт;
Т – қамтаманың құрылу уақыты, сағ.;
Ц - кВт/сағ құны, тг.

4.2 кесте – Технологиялық мақсаттар үшін электр энергиясының шығындары

| Компьютердің қуатын тұтыну, кВт / сағ | 1 кВт құны, тг. | Бағдарламаны әзірлеу уақыты, сағ. | Бағдарламаны әзірлеу уақыты, күн | Электр энергиясының жалпы шығыны, тг. |
|---|---|---|---|---|
| 0,135 | 19,17 | 232 | 29 | 600,4 |

## 4.4 Бағдарламалық өнімді әзірлеуге кеткен шығындарды есептеу

Жобалық шешімді әзірлеуге толық шығындарды есептеу мынадай формула бойынша жүзеге асырылады:

$$C_{ni} = З_{тр} + З_{сзі} + М_i + Р_{сі} + П_{зі} + Р_{нi}, \quad (4.2)$$

мұнда, $З_{тр}$ – еңбекақыға кеткен шығындар, теңге;
$З_{сзі}$ – әлеуметтік салық бойынша аударымдар, теңге;
$М_i$ – материалдарға кеткен шығындар, теңге;
$Р_{сі}$ – әзірлеу үшін қажетті арнайы бағдарламалық құралдарға кеткен шығындар, теңге;
$П_{зі}$ – басқа шығындар, теңге;
$Р_{нi}$ – үстеме шығыстар, теңге.
Жұмысшының еңбекақы мөлшері келесі формула бойынша есептеледі:

$$З_{фот} = \sum_{i=1}^{n} ЧС_i \times T_i, \quad (4.3)$$

мұнда $T_i$ – ПҚ-ны әзірлеуге жұмсалған уақыт;
$ЧС_i$ – сағаттық мөлшерлеме, теңге/сағ.
Еңбек ақы төлеу шығындары 4.2-кестеге жазылады.

4.3 кесте - Еңбекақы төлеу шығындары

| Орындаушы | Жұмсалатын уақыт, адам х сағ. | Сағаттық мөлшерлеме, теңге/сағ | Сомасы, теңге |
|---|---|---|---|
| Әзірлеуші - бағдарламашы | 1 х 232 | 700 | 162400 |
| Еңбекақының жалпы құны | | | 162400 |

Әлеуметтік салық қызметкердің еңбекақысынан 9,5% құрайды және мына формула бойынша есептеледі:

$$З_{сзi} = (З_{тр} - ПО - ВОСМС) \times 9{,}5\% - З_{coi}, \quad (4.4)$$

Мұнда, ПО – еңбекақының 10% - ын құрайтын зейнетақы аударымдары.

$$ПО = З_{тр} \times 10\% , \quad (4.5)$$

Әлеуметтік аударымдар әзірлеушінің табысының 3,5% құрайды, мынадай формула бойынша есептеледі:

$$З_{coi} = (З_{тр} - ПО) \times 3{,}5\%, \quad (4.6)$$

Міндетті әлеуметтік медициналық сақтандыру бойынша жарналарға аударымдар (ӘМСЖА) қызметкердің еңбекақысының 2% - ын құрайды және мынадай формула бойынша есептеледі:

$$ӘМСЖА = З_{тр} \times 2\% , \quad (4.7)$$

Сонымен,

$$ПО = 162400 \times 10\% = 16240 \text{ тенге};$$
$$ӘМСЖА = 162400 \times 2\% = 3248 \text{ тенге};$$
$$З_{coi} = (162400 - 16240) \times 3{,}5\% = 5116 \text{ тенге};$$
$$З_{сзi} = (162400 - 16240 - 1624) \times 9{,}5\% - 5116 = 8616 \text{ тенге}.$$

Барлық салықтың сомасы жалақының 10,46% құрайды. Осы жерден:

$$З_{сзi} = 162400 \times 10{,}46\% = \ 16987 \text{ тенге}$$

Материалдарға кеткен шығындар келесі формуламен анықталады:

$$M_i = \frac{З_{тр} \times Н_{мз}}{100\%}, \quad (4.8)$$

Мұндағы $Н_{мз}$ - еңбекақыдан түскен материалдар шығысының нормасы (3-5%).

Сондықтан материалдарға кеткен шығындар келеседей болады,

$$M_i = \frac{162400 \times 5\%}{100\%} = 8120 \text{ тенге}.$$

Әзірлеу үшін 4,3-кестеде көрсетілген құралдар, бағдарламалық және техникалық қамтамасыз ету қажет

4.4 кесте – Бағдарламаны жасау үшін техникалық және бағдарламалық қажет құралдар

| Атауы | Сипаттамасы | Саны, дана | 1 дананың бағасы, тг | Сомасы, тг |
|---|---|---|---|---|
| ASUS x556u, ноутбук | Intel Core i7 7700U 2300 MHz / 15.6" / 1920x1080 / 8Gb DDR4 / 930 GB HDD / Intel HD Graphics 940mx / Wi-Fi / Bluetooth / Windows 10 | 1 | 432000 | 430000 |
| PyCharm | Python программалық кодын орындауға арналған әзірлеу ортасы | 1 | 8 500 | 8 500 |
| Жалпы сома | | | | 438500 |

Жылдық амортизациялық аударымдар сомасы мынадай формула бойынша азайтылатын қалдық әдісімен анықталады:

$$A = \frac{Ф \times H_a}{100}, \quad (4.9)$$

мұнда Ф – негізгі өндірістік қорлардың бастапқы құны;

$H_a$ – амортизация нормасы.

Азайтылатын қалдық әдісін пайдалану кезінде амортизацияның екі еселенген ставкасы қолданылады.

Негізгі қорлар амортизациясының жылдық нормалары ҚР Салық кодексі бойынша қабылданады немесе негізгі қорларды пайдалы пайдалану мерзімінің негізінде мына формула бойынша анықталады:

$$H_a = \frac{100}{T}, \quad (4.10)$$

мұнда Т - пайдаланудың ықтимал мерзімі, жыл.

Бағдарламалық өнім үшін пайдалы пайдалану мерзімі – 4 жыл.

$$H_a = \frac{100}{4} \times 2 = 50\% \quad (4.11)$$

Амортизациялық аударымдар 4.5-кестеде келтірілген.

4.5 кесте - Амортизациялық аударымдар

| Аралық | Амортизация нормасы | Кезең ішіндегі Амортизация | Жинақталған амортизация | Жыл соңындағы баланстық құны |
|---|---|---|---|---|
| | | | | 438500 |
| 2020 ж. | 50% | 220250 | 220250 | 219250 |
| 2021 ж. | 50% | 110125 | 330375 | 109125 |
| 2022 ж. | 50% | 55062.5 | 385437.5 | 54812.5 |
| 2023 ж. | 50% | 27531.25 | 438500 | 0 |

Соңғы кезеңдегі Амортизация алдыңғы кезеңнің соңындағы баланстық құнына тең.

Бағдарламалық өнімді әзірлеуге жалпы шығындарды есептеу үшін әзірлеу кезеңі үшін амортизациялық аударымдарды есептеу қажет:

$$З_{ам} = \frac{Ф \times Н_а \times N}{100\% \times 12 \times t}, \quad (4.12)$$

мұнда N – программалық қамтаманы қолдану мерзімі, күн.

t – 1 айдың ішіндегі жұмыс күндерінің саны.

Негізгі өндірістік қорлар өнімді әзірлеу және тестілеу кезеңінде ғана пайдаланылады, бұл пайдалану уақыты 232 сағатқа немесе 29 күнге тең.

Демек,

$$З_{ам} = \frac{438500 \times 50\% \times 29}{100\% \times 12 \times 21} = 25433 \text{ тенге}. 6358250$$

"Өзге шығындар" бабы басқару аппаратын, қосалқы шаруашылықтарды және тәжірибелік (эксперименталдық) өндірістерді ұстауға арналған шығындарды, сондай-ақ жалпы шаруашылық мұқтаждықтарға арналған шығыстарды қамтиды, орындаушылардың еңбекақысына пайыздық қатынаста нормативтік бойынша нақты шығыстарға жатады. Норматив тұтастай ұйым белгілейді:

$$P_{нi} = З_{тр} \times \frac{Н_{нр}}{100}, \quad (4.13)$$

мұнда $P_{нi}$ – нақты бойынша үстеме шығындар (мың теңге);

$Н_{нр}$ – жалпы ұйым бойынша үстеме шығыстар нормативі 70% - ға тең.

Демек,

$$P_{\text{н}i} = 162400 \times \frac{70}{100} = 113680 \text{ тенге.}$$

"Өзге шығындар" бабы бойынша нақты БҚ-ға арналған шығыстар арнайы ғылыми-техникалық ақпарат пен арнайы әдебиетті сатып алуға және дайындауға арналған шығындарды қамтиды. Жалпы ұйым бойынша әзірленетін норматив бойынша, жалақыға пайызбен анықталады:

$$\text{П}_{\text{з}i} = \text{З}_{\text{тр}} \times \frac{\text{Н}_{\text{рнк}}}{100}, \quad (4.14)$$

мұнда $\text{Н}_{\text{рнк}}$ – ұйым бойынша жалпы өзге шығындар нормативі, 20% - ға тең.

Демек,

$$\text{П}_{\text{з}i} = \text{З}_{\text{тр}} \times \frac{20}{100} = 32480 \text{ тенге.}$$

Орындалған есептеулердің нәтижелері 4.6 кестеде жазылған.

4.6 кесте – Бағдарлама үшін есептелген шығындар

| Шығындар | Шартты белгіленуі | Мәні, тенге | Жалпы сомадан пайызы |
|---|---|---|---|
| Еңбекақы | $З_{тр}$ | 162400 | 20,6% |
| Еңбекақыдан алынатын салықтар | $Зн$ | 16987 | 2,2% |
| Материалдар | $M_i$ | 8120 | 1,03% |
| Арнайы жабдықтар | $P_{сі}$ | 440500 | 55.9% |
| Электр энергиясына кеткен шығындар | $З_Э$ | 600 | 0,1% |
| Негізгі қорлардың амортизациясы | $З_{ам}$ | 13142 | 1,67% |
| Басқа шығындар | $П_{зі}$ | 32480 | 4,12% |
| Үстеме шығыстар | $Р_{нi}$ | 113680 | 14,43% |
| Барлығы | $С_{пі}$ | 787909 | 100% |
| Еңбекақыдан алынатын салықтар | $Зн$ | 16987 | 2,2% |
| Материалдар | $M_i$ | 8120 | 1,03% |
| Арнайы жабдықтар | $P_{сі}$ | 440500 | 55.9% |
| Электр энергиясына кеткен шығындар | $З_Э$ | 600 | 0,1% |
| Негізгі қорлардың амортизациясы | $З_{ам}$ | 13142 | 1,67% |
| Басқа шығындар | $П_{зі}$ | 32480 | 4,12% |
| Үстеме шығыстар | $Р_{нi}$ | 113680 | 14,43% |
| Барлығы | $С_{пі}$ | 787909 | 100% |
| Еңбекақыдан алынатын салықтар | $Зн$ | 16987 | 2,2% |

Өнімнің бастапқы бағасын мына формула бойынша есептеуге болады:

$$\mathrm{C_0} = C_{ni} + p, \quad (4.15)$$

$\mathrm{C_0}$ – табыс

$C_{ni}$ – өнімнің өзіндік бағасы

р – күтілетін пайда (шамамен өнімнің өзіндік бағасының 20% - 40% құрайды)

$$p = C_{ni} * 0.2 = 787909 * 0{,}4 = 315163{,}6$$

$$\mathrm{C_0} = C_{ni} + p = 787909 + 315163{,}6 = 1103072{,}6$$

12% тең ҚҚС есебімен дайын өнімнің бағасы мынадай формула бойынша есептеледі:

$$\mathrm{C_р} = \mathrm{C_0} + \mathrm{НДС}, \quad (4.16)$$

Осыдан, бағдарламалық өнімнің қорытынды бағасы төменгі мәнге тең:

$$\mathrm{C_р} = 1103072{,}6 + (1103072{,}6 \times 0.12) = 1235441{,}3 \text{ тенге.}$$

## 4.4 Бағдарламалық өнімнің салыстырмалы экономикалық тиімділігін есептеу

Матчтың қорытындысын болжай алатын қарапайым нейрожелі спорт талдаушыларының құрамын толығымен алмастыра алмайды, өйткені спорттық ойынның нәтижесіне әсер ететін факторлар тым көп. Және нейрожеліге жаңа факторларды енгізіп, оны жаттықтыруға уақыт жетпейді.

ПҚ-ны енгізуден күтілетін жылдық әсердің шамасы мына формула бойынша есептеледі

$$\mathrm{Э_г} = \mathrm{Э_{уг}} - \mathrm{К} \times \mathrm{E_н}, \quad (4.17)$$

где $\mathrm{Э_г}$ – күтілетін жылдық экономикалық тиімділік, тенге;

$\mathrm{Э_{уг}}$ – күтілетін шартты-жылдық үнем, тенге;

К – капиталды салымдар, тенге;

$\mathrm{E_н}$ – капиталды салымдар тиімділігінің нормативтік коэффициенті.

$\mathrm{E_н}$ келесі формула бойынша анықталады:

$$\mathrm{E_н} = \frac{1}{\mathrm{T_н}} = 0.25 \quad (4.18)$$

мұнда $Т_н$ – капиталды салымдардың өтелімділігінің нормативтік мерзімі, жыл.

Бағдарламалық өнімдер үшін өтелімділік мерзімі 4 жылға тең.

Спорт аналитигінің алатын орташа төлемақысы 150 000 тг-ге тең. Букмекерлік кеңседе спорт аналиктерінін қанша адам болатынын айту қиын, шағын кеңсеге 1 адам да жетуі мүмкін. Бір адамның жылдық жалақысы 1800000 теңгеге тең болады.

Сонда,

$$Э_г = 1800000 - 1235441,3 * 0.25 = 1491139,7 \text{ тг}$$

Капиталдық салымдардың экономикалық тиімділігінің есептік коэффициенті:

$$Е_р = \frac{Э_{уг}}{К}, \quad (4.19)$$

Капиталдық салымдардың өтелімділігінің есептік мерзімі мына формула бойынша жүргізіледі:

$$Т_р = \frac{1}{Е_р}, \quad (4.20)$$

Осы жерден,

$$Е_р = \frac{1491139,7}{1235441,3} = 1,21;$$

$$Т_р = \frac{1}{1,21} = 0,82 \text{ жыл.}$$

Өтімділік мерзімі 0,82 жылды құрайды немесе 9 айдан астам.

Есептеулердің нәтижесі төмендегі кестеде көрсетілген.

4.7 кесте – Капиталдық салымдардың экономикалық тиімділігі

| Көрсеткіштердің атауы | Мәні |
|---|---|
| Шығындарды шартты жылдық үнемдеу, теңге | 1800000 |
| Капиталды салымдардың экономикалық тиімділігінің коэффициенті | 1,21 |
| Капиталды салымдардың өтелу мерзімі, жыл | 0,82 |

## 4.5 Экономикалық бөлім бойынша қорытынды

Бұл бағдарламаның ерекшелігі – бұл қызметкерлерді бақылау және жұмыс уақытын бақылау бөлімдерінде қажетті мәліметтерді есепке алудың қарапайым және икемді түрі. Қатысушыларды есепке алу бағдарламасы дәл

мәліметтермен жұмыс істеуге мүмкіндік береді, яғни сіз қате анализдермен немесе есептеулермен компанияға зиян келтіре алмайсыз. Электрондық есеп жүйесі ондағы әсерлі ақпараттарды өнімділікті жоғалтпай-ақ сақтау үшін жасалған. Қызметкерлердің әрқайсысы қажет болғаннан көп қол жеткізе алмайды - жұмыс уақытының есебінің бағдарламасында оған компаниядағы рөліне сәйкес келетін функциялар, мүмкіндіктер мен модульдер ғана қол жетімді болады.

## 2 Тіршілік қауіпсіздігі бөлімі

### 5.1 Мониторлардың визуалды параметрлерін бағалау, эргономиканы қамтамасыз ететін шараларды әзірлеу

Компьютердің көрнекі эргономикалық параметрлері қауіпсіздік параметрлері болып табылады және оларды дұрыс таңдамау пайдаланушының денсаулығының нашарлауына әкеледі. Мұндай хабар әсіресе сіздің көздеріңіз екенін есте сақтаған кезде маңызды болады. Сізді күтетін қауіптер созылмалы көз ауруларының өршуімен, тұқым қуалайтын бейімділіктің көрінісімен байланысты болуы мүмкін. Сондықтан, ДК-мен жұмыс режимі, алдын-алу шаралары және, әрине, ең бастысы - бейне мониторлардың эргономикалық параметрлері.

Ақпаратты қабылдаудың ыңғайлылығының тиісті деңгейімен сенімді оқуды қамтамасыз ету үшін визуалды эргономикалық параметрлердің оңтайлы диапазондары анықталуы керек.

5.1 кесте - Әр түрлі типтегі мониторлардың салыстырмалы визуалды эргономикалық параметрлері

| Параметр атауы | Параметр мәні |
|---|---|
| Суреттің егжей-тегжейлері мен фонының контрасты.<br>Бір пиксельдік интервалмен бөлінген бір пиксель кескіні үшін контраст | 3:1 ден кем емес<br><br>1,5:1 |
| Белгінің құрылым элементтерінің біркелкі емес жарықтығы | 1,5:1 кем емес |
| Дискретті (матрицалық) экрандар белгілері элементтерінің біркелкі емес жарықтығы | ±20 % ішінде |
| Экранның жұмыс аймағының жарықтығының біркелкі еместігі | ±20 % ішінде |
| Көршілес жарықтылықты кодтау деңгейінің контрасты | 1.5:1 кем емес |
| Белгілеу сызығының салыстырмалы ені | 1/6 бастап 1/12 дейін бас әріптер биіктігі |
| Араластырылмайтын қалдық, мм:<br>орталық шеңберде диаметрі жұмыс алаңының тік жағының ұзындығына тең<br>қалған нөлдік жұмыс шегінде | 0,3 кем емес<br>0,5 кем емес |
| Уақытша кескін тұрақсыздығы (жыпылықтау) | Бекітілмеуі керек |
| Отношение яркостей в зоне наблюдения (экран, лицевая панель, дисплей қорабы, құжаттар) | 10:1 көп емес |
| Кеңістіктегі кескіннің тұрақсыздығы (діріл). Суреттің ығысу амплитудасы 0,5-30 Гц, мм жиілікте | $2x10^4xl$ көп емес (мұндағы 1 - жобалық бақылау қашықтығы, мм) |
| Матрицаның форматы: бас әріптер мен сандар үшін фракциялар үшін бір таныс және жоласты және жол астындағы мәтін | 7х9 кем емес<br>5х7 кем емес<br>4х5 кем емес |
| Бас әріптер үшін таңба енінің оның биіктігіне қатынасы | 0,7 ден 0,9 дейін(0,5 - 1,0 рұқсат етілген) |

*5.1 кестенің жалғасы*

| | |
|---|---|
| Алфавиттік шрифт үшін таңба аралығы, шығуы жоқ | Кем дегенде контур сызығының ені немесе бір пиксель |
| Сөздер аралығы | Белгі матрицасының енінен кем емес |
| Мәтін жолдарының арасы | Кемінде 1 пиксель |
| Бақылау сызығының бұрышы | Көлденеңінен 30 ° төмен емес |
| Жұмыс аймағында кескіннің бұрмалануы:<br>бағандағы іргелес белгілердің максималды көлденең еселенуі<br>жолдағы іргелес таңбалардың максималды вертикалды есебі<br>жұмыс алаңында белгілердің бірдей түрін өзгерту<br>жұмыс алаңындағы мәтін жолдарының ұзындығындағы максималды айырма<br>жұмыс алаңындағы баған ұзындығындағы максималды айырма | <br>Таңбаның енінен 5% аспайды<br><br>Таңбаның биіктігінен 5% аспайды<br><br>Белгі биіктігінен ± 5% шегінде<br>Жолдың ұзындығының 2% аспайды<br>Баған ұзындығының 2% |
| Тіктөртбұрыштан жұмыс өрісі формасының ауытқуы:<br>Көлденең: $\Delta H = 2\frac{H_1 - H_2}{H_1 + H_2}$,<br>Вертикал бойынша: $\Delta B = 2\frac{B_1 - B_2}{B_1 + B_2}$,<br>диагональ бойынша: $\Delta D = 2\frac{D_1 - D_2}{D_1 + D_2}$<br><br>(мұндағы H1 және H2 - жұмыс өрісіндегі сол жақ және оң жақ бағаналардың ұзындықтары, мм; B1 және B2 - жұмыс өрісіндегі жоғарғы және төменгі жолдардың ұзындығы, тиісінше, мм; D1 және D2 - жұмыс алаңының диагональдары, мм). | <br>0,02 көп емес<br><br>0,02 көп емес<br><br>$0{,}04\frac{H_1 + H_2}{B_1 + B_2}$<br>көп емес |

«Дән» мөлшеріне байланысты индикатор - бұл оңтайлы мәндер келесі мәндерге жетуі керек ажыратымдылық: 15 дюйм үшін - 800-600 пиксель немесе пиксель, 17 дюйм үшін - 1024′ 768 пиксель, 19 дюйм үшін - 1280′ 1024 баллдар, 21 дюйм үшін - 1600′ 1200 және т.б. Әрине, «дән» мөлшері таңдалған ажыратымдылықты сақтауға мүмкіндік беруі керек.

*Техникалық шаралары:*

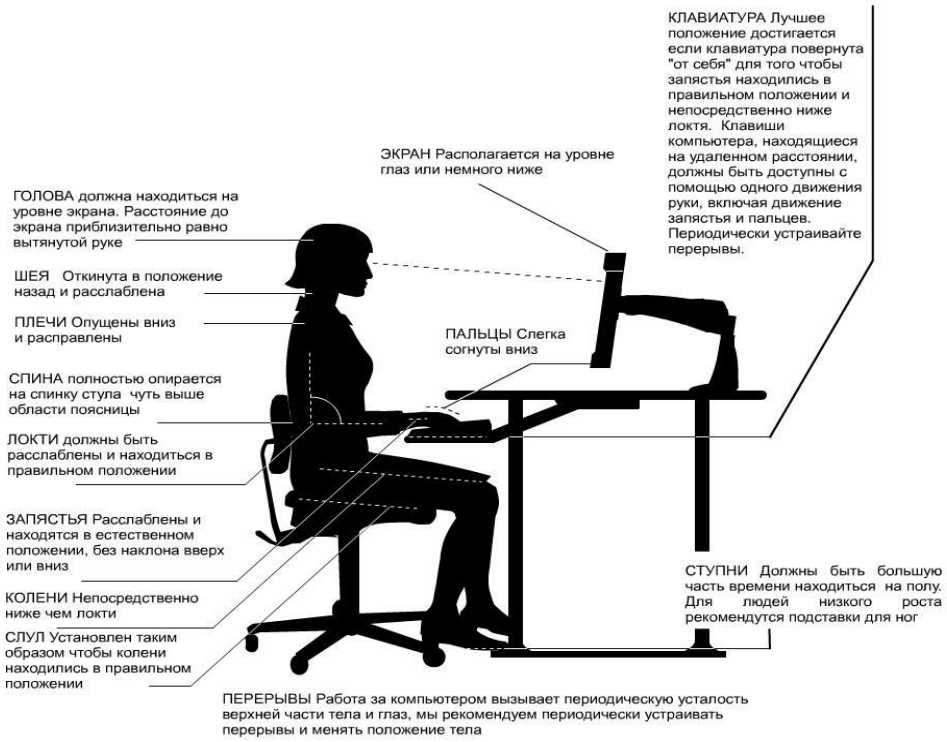
- өмірге қажетті жүйелерді (Дисплей мен компьютерге қойылатын талаптар, дисплейлер мен компьютерлердің жұмысына қойылатын талаптар, т.б.) орнатқан кездерде өрт қауіпсіздігі нормаларын сақтау;

- құрал-жабдықтар жұмысының тәртібі мен технологиялық процестер парамертлерін сақтау;

- әртүрлі қорғану жүйелерін пайдалану.

*Ұйымдастырушылық шаралар* - жұмыс орнын дұрыс үнемдеу, компьютерлерді қауіпсіз пайдалану бойынша оқу іс-шараларын өткізу, уақытылы үзілістер жасау, осылайша мониторлардың назарынан тыныштық беру.

*Жұмыс орны* - Жақсы ұйымдастырылған жұмыс орны өмірді едәуір жеңілдетеді және сізді көптеген қиыншылықтардан құтқарады. Төмендегі суреттен дұрыс сөздің мағынасы неде екенін көруге болады.Оптикалық датчиктердің плюстеріне жоғары емес баға және өртті ерте сатыда анықтау қабілеті жатады, алайда бөлмедегі шаңданудан немесе сезімтал элементтерден мұндай датчиктер жалған дабыл бере алады. Сонымен қатар, кейбір модельдер, мысалы, резеңке жану кезінде бөлінетін қара түтінге жауап бермейді.




5.1 сурет – Жұмыс орны

*Монитор* көзді негізгі тітіркендіргіш болып табылады. Мониторды таңдауға мұқият болу керек, оны үнемдеуге болмайды, өйткені сапасыз арзан монитор қысқа уақыт ішінде көру қабілетіңізді бұзуы мүмкін. Ең дұрысы, сіз IPS матрицасы және диагоналы кемінде 22 with болатын мониторларды пайдалануыңыз керек, мысалы, Apple мониторлары өте жақсы. Бюджет жеткіліксіз болған кезде арзан модельдерді таратуға болады. Мысалы, мұнда өте жақсы және салыстырмалы арзан монитор - DELL UltraSharp U2412M.

*Кесте* - бұл жұмыс кеңістігі және ол кең болуы керек. Жақсы үстелдің үстіңгі тақтайының ауданы 1 м 20 см-ден кем болмауы керек, еденнен

үстелдің биіктігі ұсынылған еуропалық стандарттарға сәйкес 74 см болуы керек, кеңсе үстелінің тереңдігі компьютер мониторының экранына дейінгі қашықтық кемінде 50 см болатындай жеткілікті болуы керек. үстел пернетақта мен тінтуірге арналған арнайы саяхат стендімен жабдықталған болуы керек.

*Орындық* жұмыс орнының негізі болып табылады, сондықтан оны белгілі бір адамның антропометриялық сипаттамаларына бейімдеу керек, яғни дененің өлшемін, салмағын және басқа да ерекшеліктерін ескере отырып түзету керек. Кафедраға қойылатын негізгі талаптар:

- тұрақты болуы керек;
- биіктігі бойынша реттелетін;
- арқаға тұрақты қолдау көрсететін арқаның болуы өте маңызды. сондай-ақ, орындықтың артқы жағында тік және көлденең түзетулер болуы керек;
- орынның беті алдыңғы жиектерімен тегіс болуы керек. Орынның ені мен тереңдігі шамамен 40 см тереңдікте;
- орындықта қолдардың тіректері, реттелетін биіктігі және аралықтары бар, олар пернетақтада теру кезінде шынтақтарды ұстап тұру үшін қажет.

Енді жұмыс орнында өзін қалай ұстау керектігін білу керек. Ең алдымен, дененің жағдайын бақылау маңызды. Ол келесідей болуы керек:

Аяқтар еденде ыңғайлы болуы керек. Бұл жұмыс орнының биіктігін реттеу арқылы мүмкін. Егер сіз қысқа болсаңыз, тіреуішті қолдана аласыз;

- аяқтардағы қан ағып кетпеуі үшін тізелер денеге кедергі келтіретін бұрыш жасауы керек;
- білектер мен шынтақтарды босаңсытып, табиғи жағдайға келтіріп, салмақ көтермеу керек. ең дұрысы, білек пен қолдың арасында 90 ° бұрыш болуы керек.
- ішкі мүшелерді қысып алмау және олардың қалыпты жұмыс істеуін қамтамасыз ету үшін тік отыру керек немесе орындықтың артқы жағын сәл қисайту керек;
- аяқтарыңыз бен артқы жағыңыздың ауырып қалуын болдырмау үшін орынның барлық тереңдігін пайдалану керек.өрт сигнализациясының түрін таңдау кезінде өрт қауіпсіздігі жүйесінің құрамында абқ орындауға тиіс міндеттерді, сондай-ақ қорғау объектісіндегі оның жұмысының техникалық шарттарын басшылыққа алу қажет.

5.2 сурет – Жұмыс орнының сипаттамасы

Есіңізде болсын, сіз ыңғайлы жұмыс орнын ұйымдастырсаңыз да, сіздің денеңіз ұзақ уақыт бойы бір қалыпта болса, сіз бұлшықет ыңғайсыздығын сезіне аласыз. Сондықтан позицияңызды өзгерту керек - созыңыз, аяқтарыңызды созыңыз, арқаңызды бүгіңіз, орындықта серпіліңіз. Сіз бірнеше қарапайым жаттығулар жасай аласыз - басын жан-жағына еңкейтіп, иықтарыңызды жоғары-төмен немесе шеңбермен айналдырып, денеңізді алға-жағына қисайтыңыз.

Сондай-ақ, мезгіл-мезгіл тұрып, кеңседе серуендеген пайдалы, әйтпесе сыртта жүру керек. Бұл күлкілі емес, бірақ темекі шегушілер темекі шекпейтіндерге қарағанда никотиндік аштық оларды жиі шығарады. Алайда, бұл темекі шегуді бастауға себеп емес, сіз ауамен демалу үшін компанияға үйлене аласыз. Немесе сіз жай серуендеуге баруға болады. Маған сеніңіз, шамамен 15-20 минуттық жылдамдықпен жүріңіз, аяғыңызды өте жақсы созыңыз және компьютерде жұмыс істегеннен кейін миды тазартыңыз.

Жеке тақырып - бұл көздер. Олар жұмыстан қатты шаршайды, демалу керек. Көзді босаңсыту үшін көруді ауыстыру өте пайдалы: қашықтықты бес минутқа қарап қойыңыз немесе бір-екі минутқа демалу үшін көзіңізді жабыңыз.

Сондай-ақ, көз гимнастикасын жасауға болады:

- көздің солдан оңға қарай көлденең қозғалысы;
- көз алмаларының тігінен жоғары және төмен қозғалысы;
- көздің айналмалы қозғалысы сағат тілімен және қарама-қарсы бағытта;
- жылдам қарқынмен көзді қатты сығу және тарту;
- көздің диагональды қозғалысы: көзіңізді сол жақ төменгі бұрышқа бұрыңыз, содан кейін тіке жоғары қараңыз. сол сияқты қарама-қарсы бағытта.
- мұрынға көздің түсуі;
- көздердің жиі жыпылықтауы;
- «қашықтықта» көз жұмысы. біз терезеге жақындап, мұқият, айқын көрінетін детальға мұқият қараймыз: терезенің сыртында өсетін ағаш бұтағы немесе әйнектегі сызаттар. Содан кейін біз ең алыс нысандарды көруге тырысып, қашықтыққа қараймыз.

Барлық кешен 5 минутты алады. Мұны әр 3-4 сағат сайын орындау керек, яғни. жұмыс күні ішінде шамамен 2-3 рет. Әр жаттығуды әр бағытта кемінде 6 рет қайталау керек. Үнемі орындау арқылы сіз көзіңізді сақтап қана қоймай, сонымен бірге көруді де жақсарта аласыз.

### 5.2 Жабық желдету жүйесінің есебі

Желдету қолайлы климатты құрудың негізгі элементі болып табылады, ол көшеден таза ауаны шығаруға және ластанған ауаны үй-жайдан шығаруға арналған.

Бөлмедегі ауа денсаулыққа, демек, осы бөлмелердегі адамдардың жұмыс істеу қабілетіне әсер ететін маңызды фактор болып табылады.

Желдету - адам өмірінің қалыпты жағдайын қамтамасыз етудің маңызды жүйелерінің бірі. Егер ол басқа климаттық жүйелермен үйлесетін болса, онда бөлмелерде жайлы микроклимат сақталады. Желдету - бұл құрылыс нормаларына сәйкес бөлмедегі және жұмыс орындарындағы ауа ортасының берілген жағдайын қамтамасыз ету үшін ауа алмасуды ұйымдастыруда қолданылатын шаралар мен құрылғылар жиынтығы. Біз бөлмеге кіруі керек таза ауа туралы сөйлесіп жатырмыз. Бұл үшін үй-жайларда желдету жүйелері орнатылады.

Барлық ғимараттарда орталық желдеткіш көтергіштері бар, әр қабатта бұтақтар бар, ол арқылы ас үй мен ванна бөлмесінен табиғи сығындылар алынады, нәтижесінде бөлмедегі ең қарапайым табиғи ауа алмасуы ұйымдастырылады: ауа желдеткіш торлардан шығып, біртіндеп терезелер, есіктер, әр түрлі ағулар арқылы көшеге шығады. буындар және т.б. Әр түрлі мақсаттағы үй-жайларды пәтерлерден өндірістік үй-жайларға дейін желдету мәселелерін шешу үшін көптеген желдету жүйелері бар, оларда ауа айналымының қажетті көлемін әр түрлі қуаттылықтардың желдеткіштері қамтамасыз етеді, осы жүйелерден басқа, әдетте, ауаны тазартудың қосымша бөлімдері бар: жылыту, сүзу, ылғалдандыру, салқындату және т.б. қажеттілік бойынша.

Солай ауадағы ауа алмасудың қажетті жылдамдығын анықтайық.

Біздің бөлмеде 3 адам жұмыс істейдң

5.2 кесте - Әр түрлі жұмыс кезінде адам шығаратын көмірқышқыл газының мөлшері

| Возраст человека и характер работы | Количество $CO_2$ | |
|---|---|---|
| | в л/ч | в г/ч |
| Взрослые: | | |
| при физической работе | 45 | 68 |
| при легкой работе (в учреждениях) | 23 | 35 |
| в состоянии покоя | 23 | 35 |
| Дети до 12 лет | 12 | 18 |

5.3 кесте – Көмірқышқыл газының шекті рұқсат етілген концентрациясы

| Наименование помещений | Количество $CO_2$ | |
|---|---|---|
| | в л/м$^3$ | в г/кг |
| Для постоянного пребывания людей (жилые ком.) | 1 | 1,5 |
| Для пребывания детей и больных | 0,7 | 1 |
| Для учреждений | 1,25 | 1,75 |
| Для кратковременного пребывания людей | 2 | 3 |

5.2 кестеге сәйкес қосымшасында 1 шығарылған CO2 мөлшерін анықтаңыз.

бір адам г = 23 л / сағ. 2 кестеге сәйкес 1CO2 рұқсат етілген концентрациясын анықтаймыз. Сонда xv = 1 л / м3

және үлкен қалаларға арналған ауадағы CO2 мөлшері xn = 0,5 л / м3

L = 23 · 3 / (1 - 0.5) = 138 м3 /сағ Жауабы: L = 138 м3/сағ

### 5.3 Өндірістік бөлменің жасанды жарықтандыру жүйесін жасау

Қолданыстағы арматураның сызбасын құру;

нүкте әдісін қолдана отырып, жұмыс орнындағы жарықтандыруды есептеңіз;

жарықтандырудың стандартты мәндерін ҚР ҚНжЕ 2.04-05-2002 сәйкес келтіріңіз. Табиғи және жасанды жарықтандыру. Жалпы талаптар;

өндірістік бөлменің жарықтандыру жүйесін қайта құру арматураның жоспарланған орналасуын келтіріңіз;

қорытындылар мен ұсыныстар жасау.

Көрнекі жұмыстардың категориясы I (а), сондықтан кестеге сәйкес стандартты жарықтандыру En = 500 люкс.

Нүктелік әдіс бойынша берілген шамдар мен шамдардың стандартталған мәнге сәйкестігін тексереміз.

Определение расчетной высоты подвеса:

$$h_1 = H - h_{\text{р.п.}} - h_{\text{с.в.}} = 5 - 1 - 1 = 3 \text{ м} \qquad (1)$$

мұндағы Н - бөлменің биіктігі;

$h_{р,п}$ - дүкендегі жұмыс бетінің биіктігі, (0,8 ÷ 1 м);

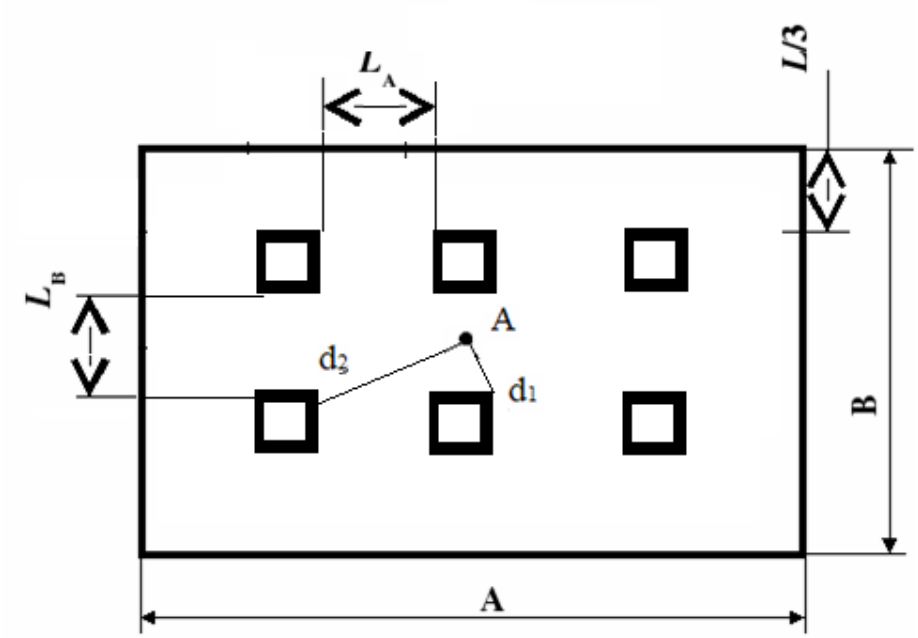
$h_{св}$ - шамның шамадан тыс биіктігі (0 ÷ 1,5 м).

| Сила света Iа, кд в направлении а , градусов | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Тип свет-ка ПВЛМ-1 х 40 | 0 | 5 | 15 | 25 | 35 | 45 | 55 | 65 | 75 | 85 | 90 |
| | 139 | 135 | 132 | 115 | 104 | 84 | 63 | 44 | 22 | 6 | 0 |

Таблица 1 - Светораспределение светильника ПВЛМ-2 х 40

Таблица 2

| | d ,м | | ,град | ,кд | л.к. |
|---|---|---|---|---|---|
| 0 | 2 | 1,00 | 45 | 84,00 | 7,42 |
| 1 | 4,47 | 2,24 | 65,90 | 42,02 | 0,72 |
| 2 | 7,21 | 3,61 | 74,49 | 23,12 | 0,11 |
| 3 | 10,77 | 5,39 | 79,48 | 13,17 | 0,02 |
| 4 | 6,00 | 3,00 | 71,57 | 36,45 | 0,29 |
| 5 | 10,00 | 5,00 | 78,69 | 11,90 | 0,02 |




5.3 сурет – Жарық орналастыру схемасы

Шамдар ара қашықтығы ($L_{A,B}$):

$$L_A = \lambda \cdot h_1 \quad (2)$$

$$L_B = \lambda \cdot h_1 \quad (3)$$

мұнда λ – техникалық тұрғыдағы ең тиімді ара қашықтық (λ =1,2÷2,4).

LA шамдарының арасындағы қашықтықты табу үшін бөлменің ұзындығын 3-ке бөлу керек:

$$L_A = \frac{14}{3} = 4{,}6 \text{ м}$$

Коэффициенттің жағдайын тексеру λ=1,2÷2,4:

$$\lambda = \frac{4,6}{4} = 1,16 - \text{қанағаттандырады}$$

$L_B$ шамдары арасындағы қашықтықты табу үшін бөлменің енін 2-ге бөлу керек:

$$L_B = \frac{8}{2} = 4 \text{ м}$$

Коэффициенттің жағдайын тексеру λ=1,2÷2,4:

$$\lambda = \frac{4}{4} = 1 - \text{қанағаттандырады}$$

Есептеу үшін біз А бақылау нүктесін белгілейміз. D1, d2 - А нүктесі мен сәйкес шам арасындағы төбеге дейінгі қашықтықтың проекциясын табу керек.

$$d_1 = \sqrt{(\frac{L_A}{2})^2 + (L_B)^2} = \sqrt{(\frac{4,6}{2})^2 + (4)^2} = 4,61 \text{ м} \quad (4)$$

$$d_2 = \sqrt{(\frac{3 \cdot L_A}{2})^2 + (L_B)^2} = \sqrt{(\frac{3 \cdot 4,6}{2})^2 + (4)^2} = 7,97\text{м} \quad (5)$$

Әрі қарай, төбенің биіктігі мен тиісті d кесіндісінің арасындағы бұрышты анықтаймыз:

$$tg\alpha_1 = \frac{d_1}{h_1} = \frac{4.61}{4} = 1.1525 \rightarrow \alpha_1 = 49.05° \quad (6)$$

$$cos^3\alpha_1 = 0.2815$$

$$tg\alpha_2 = \frac{d_2}{h_1} = \frac{7.97}{4} = 1.9925 \rightarrow \alpha_2 = 63.34° \quad (7)$$

$$cos^3\alpha_2 = 0.8679$$

- Осы бұрыштан біз 1.2 кестеге сәйкес әр көзден жарықтың арқындылығын табамыз:

- $I_1$ = 74 кд ( α = 49,05°);
- $I_2$ = 46,5 кд ( α = 63,34°).

- Әр көзден басқару пунктіне қатысты бөлмені жарықтандыру:

- $E_1 = \frac{I_1 \cdot cos^3\alpha_1}{h_1^2} = \frac{74 \cdot 0,2815}{16} = 1,301$ лк
- $E_2 = \frac{I_2 \cdot cos^3\alpha_2}{h_1^2} = \frac{46,5 \cdot 0,8679}{16} = 2,521$ лк

Жалпы жарықтандыру:

$$E_1 = \frac{\mu \cdot F}{1000 \cdot K_3} \cdot \sum_{i=1}^{6} \quad E_i = \frac{1{,}1 \cdot 3120}{1000 \cdot 1{,}2} \cdot 3{,}204 = 9{,}163 \text{ лк}$$

мұндағы μ - «қашықтан» шамдардың әсерін ескеретін коэффициент (1,1 ÷ 1,2).

## Қорытынды

Қорытындылай келе, бұл бағдарлама қызметкерлердің келу және келу уақытын бақылауға мүмкіндік береді. Ол кестеде келіп кету уақытын көрсетеді, онда әкімшілік пайдаланушылардың уақыт кестесін көре алады, яғни жай пайдаланушылар үшін есепшот бар. Осы бағдарламаны әзірлеу кезінде біз бағдарламалау тілдеріменжұмыс істеу үшін көмекші бағдарламаны, сондай-ақ жергілікті хостта дерекқор құру үшін көмекші бағдарламаны қолдандық, сол арқылы біз осы бағдарламалармен қолдануды үйреніп толықтай жұмыс істейтін бағдарламаны жаздық. Менің бағдарламам көптеген компанияларға көмектеседі деп үміттенемін, бірақ бұл өте қарапайым бағдарлама сияқты, бірақ ол өз міндетін жақсы атқарады. Бұл бағдарлама сағат ақы төлейьін компанияларға өз жұмысшыларына атқармаған жұмыс үшін төйленетін артық шығындардан қорғау керек, яғни көп жұмыс орындарында өз жұмыс орындарында кешігіп немесе ерте кететіндер болады, әрине ондай адамдар аз бірақ олар кішігірім болсада олар бар, енді осындай жұмысшыларды табу үшін менім бағдарламам компанияларға көмегін көрсететінін үміт етемін. Оның қарапайым интерфейсі қолданушыларға ұнайтынына сенемін. Қорытындылай келе, осы бағдарламаны жазу барысында мен с # тілдерімен жұмыс істеуді үйрендім, бұл WindowsForms негізінде MySql тілінде мәліметтер базасын құруға және басқаруға арналған жергілікті хосттарды құруға және басқаруға арналған қосалқы бағдарламалармен жақсы таныстым.

## Әдебиеттер тізімі


1 Шумаков, П.В. Delphi 3 и разработка приложений баз данных / П.В. Шумаков. - М.: Нолидж, 2010. - 704 ;

2 Харрис Энди PHP/MySQL для начинающих; КУДИЦ-Образ - М., 2016. - 384 с;

3 Никсон Робин Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript и CSS; Питер - М., 2017. - 204 ;

4 Кузнецов Максим , Симдянов Игорь Самоучитель PHP 5; БХВ-Петербург - М., 2017. - 560 с;

5 Зандстра Мэтт PHP. Объекты, шаблоны и методики программирования; Вильямс - М., 2016. - 560 с;

6 Дронов В. PHP, MySQL и Dreamweaver. Разработка интерактивных Web-сайтов; БХВ-Петербург - М., 2016. - 480 с.

7 Кузнецов Максим , Симдянов Игорь Самоучитель PHP 5; БХВ-Петербург - М., 2017. - 560 с.

8 Скляр Дэвид , Трахтенберг Адам PHP. Рецепты программирования; Питер - М., 2017. - 784 с.

9 Фленов Михаил PHP глазами хакера; БХВ-Петербург - М., 2016. - 991 с.

10 Албахари, Джозеф C# 3.0. Справочник / Джозеф Албахари , Бен Албахари. - М.: БХВ-Петербург, 2012. - 944 с.

11 Рихтер, Джеффри CLR via C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C# / Джеффри Рихтер. - М.: Питер, 2013. - 928 с.

12 Мак-Дональд, Мэтью Silverlight 5 с примерами на C# для профессионалов / Мэтью Мак-Дональд. - М.: Вильямс, 2013. - 848 с.

13 Ишкова, Э. А. Самоучитель C#. Начала программирования / Э.А. Ишкова. - М.: Наука и техника, 2013. - 496 с.

14 Зиборов, В.В. Visual C# 2012 на примерах / В.В. Зиборов. - М.: БХВ-Петербург, 2013. - 480 с.

15 Вагнер, Билл C# Эффективное программирование / Билл Вагнер. - М.: ЛОРИ, 2013. - 320 с.

16 Лотка, Рокфорд C# и CSLA .NET Framework. Разработка бизнес-объектов / Рокфорд Лотка. - М.: Вильямс, 2010. - 816 с

17 Фримен, Адам ASP.NET MVC 3 Framework с примерами на C# для профессионалов / Адам Фримен , Стивен Сандерсон. - М.: Вильямс, 2011. - 672 с.

18 Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 с.