ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.Сәтбаев атындағы қазақ ұлттық техникалық зерттеу университеті

Ақпараттық және телекоммуникациялық технологиялар институты

Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

Кәкіш Ерасыл Ринатұлы

Жеке тұлғаны тануға арналған веб-қосымшаны әзірлеу

ДИПЛОМДЫҚ ЖҰМЫС

5В070300 – «Ақпараттық жүйелер» мамандығы

Алматы 2021

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.Сәтбаев атындағы қазақ ұлттық техникалық зерттеу университеті
Ақпараттық және телекоммуникациялық технологиялар институты
Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ
КАӨжС кафедра меңгерушісі,
тех.ғыл.канд, ассоц. доцент
____ Н.А.Сейлова
«_31_»__05___2021 ж.

ДИПЛОМДЫҚ ЖҰМЫС

Тақырыбы: «Жеке тұлғаны тануға арналған веб-қосымшаны әзірлеу»

5В070300 – «Ақпараттық жүйелер» мамандығы

Орындаған: Кәкіш Е.Р.

Ғылыми жетекші: лектор, магистр
Дүйсенбаева А.Н.

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

Қ.Сәтбаев атындағы қазақ ұлттық техникалық зерттеу университеті

Ақпараттық және телекоммуникациялық технологиялар институты

Киберқауіпсіздік, ақпаратты өңдеу және сақтау кафедрасы

БЕКІТЕМІН
КАӨС кафедра меңгерушісі,
тех.ғыл.канд, ассоц. доцент
_____ __ Н.А.Сейлова
«__31__»__05___2021 ж.

.

Дипломдық жұмысты орындауға
ТАПСЫРМА

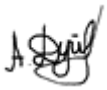
Білім алушы: Кәкіш Ерасыл Ринатұлы
Тақырыбы: «Жеке тұлғаны тануға арналған веб-қосымшаны әзірлеу»
Университет Ректорының 2020 жылғы «24» қараша №2131-б бұйрығымен бекітілген
Аяқталған жұмысты тапсыру мерзімі 2021 жылғы «20» мамыр
Дипломдық жұмыстың бастапқы берілістері: диплом алдындағы практикалық жұмыс қорытындысы, тақырып бойынша әдебиеттерге шолу нәтижелері,теориялық мәліметтердің жиыны
Дипломдық жұмыста қарастырылатын мәселелер тізімі:
а)тақырып аясын талдау және шолу
ә)адамдарды тану арқылы электрондық қатысу жүйесін әзірлеу
б)программалық қамтаманы құру
Сызбалық материалдар тізімі: Power Point бағдарламасындағы слайдтар
Сызба материалдар: 15 слайдпен көрсетілген
Ұсынылатын негізгі әдебиет: 10 атау

Дипломдық жұмысты дайындау

КЕСТЕСІ

Бөлімдер атауы, қарастырылатын мәселелер тізімі	Ғылыми жетекші мен кеңесшілерге көрсету мерзімдері	Ескерту
Мәселенің қазіргі жағдайына шолу және оны талдау	9.02.2021	
Ақпараттық қамтаманы құру	21.03.2021	
Программалық қамтаманы құру	29.04.2021	

Дипломдық жұмысының бөлімдерінің кеңесшілері мен норма бақылаушыларының аяқталған жобаға қойған **қолтаңбалары**

Бөлімдер атауы	**Кеңесшілер, аты, әкесінің аты, тегі (ғылыми дәрежесі, атағы)**	**Қол қойылған күні**	Қолы
Норма бақылаушы	Дүйсенбаева А.Н. лектор, магистр		
Программалық қамтама			

Ғылыми жетекші _____  _____Дүйсенбаева А.Н.

Тапсырманы орындауға алған білім алушы  Кәкіш Е.Р.

Күні « 24» 11 2020 ж.

АҢДАТПА

Адамдардың келбетін анықтау жүйесі – әр-түрлі салалардағы қауіпсіздік немесе басқа да жүйелерді және ақылды жасанды зерденің дамуына біршама мүмкіндіктер береді. Осы жүйені биометриялық анықтауларға, қауіпсіздік жүйелері ретінде бейне бақылаулар үшін пайдалану кең спектрде қолданылады. Бұл дипломдық жұмыс қауіпсіздік жүйесі ретінде және IOT технологияларымен байланыстырып жасауға болатын техникалық жабдықтарды қарастырады. Адамның келбетін анықтау үшін көздердің арасындағы ара-қашықтық, мұрын тесіктерінің ені мен мұрын ұзындығы, бет сүйектерінің биіктігі мен формасы, иектің ені, маңдайдың биіктігі және тағы басқалары. Бұл жұмыста – адамның келбетін анықтау үшін дайын алгоритмдері бар MEGVII Қытай компаниясының тегін Face^{++} (FacePlusPlus) сервисын және Microsoft компаниясының шығарған C# .NET Windows Form жүйесі арқылы іске асыру ұсынылады.

АННОТАЦИЯ

Системы распознавания человеческих лиц обеспечивают безопасность или другие системы в различных областях и предоставляют значительные возможности для развития интеллектуального искусственного интеллекта. Широко применяется использование данной системы для биометрической идентификации, как охранной системы для видеонаблюдения. В этой дипломной работе рассматривается техническое оборудование, которое может быть создано как система безопасности в сочетании с технологиями IoT. Расстояние между глазами, ширина ноздрей и длина носа, высота и форма скул, ширина подбородка, высота лба и многое другое определяют внешний вид человека для распознования компьютером. В данной работе предлагается реализовать бесплатный сервис Face ++ (FacePlusPlus) от китайской компании MEGVII с готовыми алгоритмами определения внешности человека и C# .NET Windows Form от Microsoft.

ABSTRACT

Systems of recognition of human beings ensure safety or other systems in different areas and provide significant opportunities for the development of intellectual and artificial intelligence. The use of this biometric identification system as a video surveillance system is widely used. In this thesis is considered technical equipment, which can be created as a security system in combination with IoT technology. The distance between the eyes, the width of the nose and the length of the nose, the height and shape of the cheekbones, the width of the chin, the height of the forehead and many others determine the appearance of a person for computer recognition. In this work it is proposed to implement a free service Face ++ (FacePlusPlus) from the Chinese company MEGVII with ready-made algorithms for determining the appearance of man and C # .NET Windows Form from Microsoft.

МАЗМҰНЫ

КІРІСПЕ

Бұл дипломдық жұмыста адамның келбетін камерада анықтап, суретке түсіріп, сол бейнелерді салыстыру арқылы адамды идентификациялау веб-қосымшасы ұсынылады. Адамның келбетін анықтау және идентификациялау үшін арнайы FacePlusPlus сервисін қолданамын. Бұл сервис адамның келбетін – көздердің ара-қашықтығы, бет пішіні, ерін мен мұрынның орналасуы және де әр-түрлі мимикалық әсерлерді бақылай отырып, өзінің алгоритмдері арқылы анықтайды.

Біз әлемдегі миллиондаған адамдарға жағымды әсер ете алатын тамаша идеяларды ұсыну үшін эксперимент жасау мен жаңалық енгізу бұрынғыдан да оңайырақ болатын интернет - революция дәуірінде өмір сүріп жатырмыз.

Сіздің сөреңізге, жәшіктеріңізге, шкафтарыңызға немесе үй есіктеріңізге қосымша қауіпсіздік қосқыңыз келді ме? Осындай қауіпсіздікті қамтамасыз ететін жүйені дайындағым келіп, дипломдық жұмыстың тақырыбы ретінде таңдадым.

Мен әзірлеген адамның келбетін анықтау веб-қосымшасының басты мақсаты – жеке сөре, шкаф қоймасынн, үйіміздің сыртқы есігін адамның келбетін анықтау арқылы есіктің құлпын ашуға мүмкіндік беру немесе арнайы офистік жерлерде жұмыскерлер үшін есікті автоматты түрде ашу және қай уақытта кіріп – шыққанын қадағалай отырып, жұмыскерлердің дисциплинасын бақылау жүйесін автоматтандыру веб-қосымшасын жазу.

Адамның келбетін анықтау жүйесін қолдану – казіргі заманызмызда көбінесе бейнебақылау жүйелерінде, қол жетімділікті басқаруда, әртүрлі мобильді және бұлтты платформаларда қолданылады.

Бұл дипломдық жұмыста келбетті анықтау алгоритмдерін және осындай арнайы қосымшаны әзірлеу - қандай иструменттер арқылы жүзеге асатынын қарастырамын.

1 АДАМНЫҢ КЕЛБЕТІН АНЫҚТАУ

1.1 Жалпы сипаттама

Адам әлемдегі нысандарды ондаған миллисекундта анықтай алады. Нысанды танудың бұл ең жоғары жылдамдығы, себебі біздің миымыз көзге көрінетін қатынастар туралы әрдайым болжамдар жасайды және осы болжамдарды қазіргі ақпаратпен салыстырады. Адамды анықтау процесінде 3 кезең бар:

– біз қарастыратын адамның физикалық ерекшеліктерін анықтау;

– біз сол адамды немесе басқа адамды білсек те, біз оны түсінетін формадағы адамның жеке басын анықтау;

– біз адамды танимыз, бірақ оның есімін білетінімізді білмейміз.

Зерттеушілер әр кезеңде мидың нақты аймақтары белсенді болатындығын анықтады. Бетті анықтау, психологтар атап өткендей, бұл когнитивті қабылдаумен тығыз байланысты процесс. Шындығында, адамды анықтау былай жүреді: ми үнемі көргенін ұзақ уақыттық жадында сақталатынымен салыстырады. Бір ғажабы, белгілі бір адамдарды анықтаудың барлық дерлік алгоритмдері, олардың мәліметтер қорына енгізілген және осылайша жұмыс істейді.

Мысалы, біз осындай нысанды бір сағатқа қараған кезде оны өзіміз көрген нәрселер мен осы сипаттамаларды салыстыру ретінде танимыз, сағаттың ақыл-ой бейнесіне кім келеді. Барлық сағаттар бірыңғай болмаса да, осы объектінің кейбір көшірмелерін ментальды прототиптен ажыратуға болатындығына қарамастан, әр сағаттарда негізгі функциялар жиынтығы бар, олардың ішінде минуттық және сағаттық қол, теру бар. Содан кейін объект суреті жіктеледі және белгілі бір категория бойынша жадыда сақталады. Сағаттың әр түрлі түрлері миға қаншалықты көп еске түсірсе, соғұрлым жаңа затты анықтау оңайырақ болады.

Объектілерді жіктеу - бұл, әдетте, тану процесі аяқталатын кезең, бірақ тұлғаны тану жағдайында бұл тек бастама. Егер сағат алуға бір сағат жеткілікті болса, онда адамның бет-бейнесі адамның бет-әлпетін анықтау үшін жеткіліксіз. Бірден дерлік біз адамның жынысы мен жасын, нәсілін, содан кейін де ол бізге ұнай ма, жоқ па, соны анықтаймыз. Сонымен қатар, біз адамды білетіндігімізді де анықтаймыз. Егер адам бізбен таныс болса, біз бұл ақпаратты, сондай-ақ тұлғаны анықтау алгоритмін шығаруды бастау үшін келдік. Ол белгілі бір адамды анықтай алады, содан кейін ақпаратқа қол жеткізе алады, өзінің жасағанына байланысты қылмыстық қызметтерге сигнал жібере алады.

1.2 Адамның келбетін машина көзімен анықтау

Беттерді анықтау және тану проблемасы - бұл объектілерді тану және

идентификациялау теориясының қалыптасуы мен дамуына түрткі болған алғашқы практикалық мәселелердің бірі. Гностикалық аймақтарға сәйкес келетін және визуалды бейнелерді тудыратын тоғыз санаттағы объектілер бар:

- манипуляциялауға болатын объектілер;
- жартылай манипуляциялауға болатын объектілер;
- объектілер манипуляцияланбайды;
- адамдар;
- мимика;
- тіршілік иелері;
- баспа белгілері;
- қолмен жазылған кескіндер;
- жарық көздерінің сипаттамалары мен орналасуы.

Бетті тану және тану процедурасына негізделген процедураларға қызығушылық әрдайым маңызды болды, әсіресе өсіп келе жатқан практикалық қажеттіліктерге байланысты: қауіпсіздік жүйелері, тексеру, криминалистика, телеконференциялар және т.б. Адамның жүзін жақсы анықтайтын күнделікті фактінің анықтығына қарамастан, бұл процедураны компьютерге қалай үйрету керек, оның ішінде беттердің сандық кескіндерін декодтау және сақтау әдісі анық емес. Беткейлердің ұқсастығын бағалау, олардың күрделі өңделуін қоспағанда, тіпті онша айқын емес. Тұлғаны анықтау проблемасын зерттеудің бірнеше бағыты бар:

- жүйке-психологиялық модельдер;
- нейрофизиологиялық модельдер;
- ақпараттық және процедуралық модельдер;
- танудың компьютерлік модельдері.

Бет-әлпетті тану проблемасы тіпті компьютерлік көрудің алғашқы кезеңінде қарастырылды. 40 жылдан астам уақыт бойы бірқатар компаниялар автоматтандырылған және қазіргі кезде адамның бет-әлпетін танудың автоматты жүйелерін дамытып келеді: Smith & Wesson (ASID - Күдіктілерді автоматты түрде анықтау жүйесі); ImageWare (FaceID жүйесі); Imagis, Epic Solutions, Spillman, Miros (Trueface жүйесі); Vissage технологиясы (Vissage галереясы жүйесі); Visionics (FaceIt жүйесі).

Бетті тану мәселесін шешу үшін әртүрлі әдістер ұсынылды, олардың ішінде Карунен - Лью кеңеюіне, алгебралық моменттерге, бірдей қарқындылық сызықтарына, серпімді (деформацияланатын) салыстыру стандарттарына негізделген нейрондық желілерге негізделген тәсілдерді бөліп көрсетуге болады. Тану алгоритмдерін жасау кезінде ерекше күштер оның әртүрлі кескіндеріндегі бет элементтерін (көз, мұрын, ауыз, иек және т.б.) автоматты түрде таңдауға бағытталған: бет, профиль және ерікті бұрыш. Әрі қарай, бұл геометриялық сипаттамалар тану мәселесін шешуде қолданылады. Бұл тәсілдерді сипаттаған кезде, беттердің статистикалық маңызды базасында салыстырудың болмауы болып табылады. Беттерді танудың екі әдісі бар:

- тітіркендіргіштер арасындағы сәйкестік түрін біреуіне қарсы салыстыру;

– адамдардың жинақталған, өкілдік саны арасындағы салыстыру.

Бет элементтерінің анықтамасына негізделген геометриялық салыстыру - Бет элементтері: көз, мұрын, ауыз, иек және т.б. Бетті жеке бет элементтері жеткіліксіз көрінген кезде де тануға болады. Тәсіл идеясы - тұлғаның жеке элементтерінің салыстырмалы орналасуы мен өзіндік сипаттамаларын табу. Бұл көрсетілді

Бет элементтері қолмен шығарылса да, компьютердің танылуы өте жақсы нәтиже береді.

Анықтамалық салыстыру байттардың массиві ретінде ұсынылған кескінді - интенсивтілік мәндерін сәйкес метрикада сілтеме - тұтас бетпен салыстырады деген ойға негізделген. Стандарттарды дайындаудың және оларды ұсынудың бірнеше әдісі бар. Әр түрлі қырынан тану үшін бірнеше стандарттар қолданылады.

Адамды әр түрлі стандарттардың жиынтығы ретінде ұсынған кездегі көзқарас назар аударарлық. Артықшылықты және жан-жақты тәсіл - бақылау бұрышын өзгерткен кезде басты тұлғаның түрленуін бағалауға мүмкіндік беретін дәл алдыңғы модельмен бірге бір сілтемені қолдану. Содан кейін деформацияланатын модель эталондық бет салыстыру метрикасын құру кезінде қолданылады. Бұл идея деформацияланатын стандарттар техникасының негізі болып табылады.

Брюс В. жұмысындағы салыстырудың салыстырмалы схемасы корреляциялық-экстремалды деп аталуы үшін жеткілікті түрде өзгертілген. Мұнда градиенттік картаға айналдыратын және шеткі картадан бос суретті қалыпқа келтіру қолданылады. Табысты нәтижелердің бірі - көзге, ауызға және мұрынға арналған бірнеше шешімдер мен шағын анықтамалық стандарттарды қолдану. Бет элементтерінің детекторлары осы тәсілдерге негізделген. Келесі қадамның сындарлы екенін атап өту маңызды: алдымен көзді анықтаңыз (салыстыру арқылы), содан кейін масштаб пен бағдар бойынша суретті автоматты түрде қалыпқа келтіріңіз.

Айта кету керек, бұл тәсіл бүкіл бет стандартына негізделген тану элементтерін қамтиды: кескінді қалыпқа келтіру үшін EL (көз) қолданылады, ал анықтамалық салыстыру тұлғаның жеке сипаттамалық ерекшеліктері үшін (көз, мұрын, ауыз). Алайда, эксперименттер көрсеткендей, тұлғаны тану бүкіл тұлғаны тану тәсілін оның элементтерін эталондық салыстыруға негізделген тәсілмен біріктіретін архитектурада сәтті болады.

Сондай-ақ негізделген тану схемасын қарастырған жөн K-L ыдырауы. K-L ыдырауындағы тану объектілері базалық стандарттардың сызықтық қосындысы түрінде көрсетілгендіктен, тану алгоритмі корреляцияға қарағанда жақсы нәтиже бере алмайтынын ескеріңіз. Алайда, осылайша EL геометриялық сипаттамаларына негізделген тану схемаларымен салыстыруға болатын есептеу шығындарын едәуір төмендетуге болады. Сонымен Эллис Х.Д. тану сапасының бірдей деңгейінде есептеу шығындарының төмендеуі 96% жететінін көрсетті. Ұқсас Т.Поджо алгоритмі Р.Д.Баронның алгоритміне қарағанда жақсы жұмыс істейді, өйткені суреттің бұрмалануына төзімді кішігірім сілтемелер

қолданылады.

Нейрондық желілерге негізделген тану схемасы қызығушылық тудырады. Атап айтқанда, 3D нысандарын ерікті бұрыштан тану үшін Face элементтерінің ерекшеліктері векторын синтездеу кезінде гипер негізді функциялар желісін қолдану. Бұл жағдайда желі элементтері - бұл Face элементтерінің параметрлері, олардың суреттегі орнын қоса. Гипербазалық функционалды желіде әрбір пиксель үшін градиенттердің амплитудасы және сәйкес стандарттардың центрлері ретінде, әртүрлі ауысымдағы әртүрлі центрлер бар, олар Face элементтерінің стандарттарын салыстыру үшін бұрын сипатталған схемаға ұқсайды. Бұл корреляция коэффициенттерін жай көбейтудің орнына Гаусс корреляция коэффициенттері бойынша сызықтық жіктеуге сәйкес келуі мүмкін.

Тану нәтижелерінің түсіру бұрышына тәуелділігі туралы мәселені бірнеше жолмен шешуге болады. Егер әр адам үшін әр түрлі қырынан түсірілген кескіндер болса, онда есептеу шығындарының артуы есебінен бірдей тану схемаларын қолдануға болады. Гипербазалы функцияларды қолдану - әртүрлі проекциялар нүктелері арасындағы интерполяция мүмкіндігімен жіктеу өте қауіпті. Алайда, шын мәнінде, сілтеме жасау үшін тек бір фронтальды бет суреті болуы мүмкін. Әрине, бір 3D кескін

объектіде (көлеңкелер жоқ) жеткілікті ақпарат жоқ. Егер, соған қарамастан, объект әртүрлі проекция нүктелері белгілі болатын ұқсас объектілердің (прототиптердің) класына жататын болса, онда ақылға қонымды экстраполяция мүмкін және тек бір 2D проекциясының көмегімен берілген объект үшін дұрыс проекцияны ұсынуға болады. Адамдар фронталь проекциядан 20-300 айналдырылған беттерді анықтай алады. Мүмкін, олар жай типтік тұлға құрылымы туралы өз нәтижелерін қолдана алады.

Бұл мәселенің тағы бір шешімі - фронтальды емес кескіндерде беттің танылуын қолдау үшін 3D бет модельдерін пайдалану проблемасы. Р.Брунелли атап өткендей, осы кластың басқа типтік объектілерінің проекциялары туралы білімді қолдана отырып, беттің басқа проекцияларын алуға байланысты сараптамалық мәліметтер базасында жұмыс жасауды қоса, есептер құрастыруға және оларды шешуге болады.

Бет элементтерінің әртүрлі контурлары үшін оларды түпнұсқалық портреттен алудың әртүрлі әдістері қолданылады. Көз бен ауыз пішіндері тұрақты геометриялық пішіндерге ие, сондықтан олар деформацияланатын эталондық модель тұрғысынан алынады. Қас, мұрын және бет контуры сияқты басқа бет ерекшеліктері соншалықты өзгермелі болғандықтан, оларды анықтау үшін белсенді контурлық модель қолданылады, ол үнемі осындай заттарды анықтайды. 3-суретте портреттік сараптамада қолданылатын және тұлғаның автоматты түрде танылуымен сәйкестендіру өте қажет болатын барлық бет элементтері көрсетілген, бұл әдіс заңдылығын қамтамасыз етеді.

Деформацияланатын стандартты модель. Деформацияланатын стандарттар EL-нің күтілетін формасы туралы априорлық біліммен белгіленген және контур декодтауымен оқу процесінде сандық түрде анықталатын параметрлермен анықталады.

Стандарттар олардың өлшемдерін және басқа параметрлерді өзгерту кезінде икемді болады, ал оларды сандық түрде салыстыруға болады, ал алынған параметр мәндерін белгілі бір сипаттамада қолдануға болады.

Бет элементтері. Деформацияланатын стандарттар ағымдағы сандық кескінмен динамикалық режимде өзара әрекеттеседі. Энергетикалық функция кескіннің бет элементтеріне сілтеме жасайтын компоненттер жиынтығымен анықталады, мысалы, биіктігі мен төмендігі, жиектері және қарқындылығы сияқты қарқындылық кесінділерінің сызбаларының сипаттамалары. Энергия функциясының минимумы берілген кескіннің ең жақсы таңдауына сәйкес келеді. Әдетте деформацияланатын стандарттар көзді және ауызды анықтау үшін қолданылады.

Бет элементтерін іздеуді бастамас бұрын, сіз бет элементтерін басқа бет бөліктерінен анықтауға болатын жарықтық шекараларын және әрбір қайталанудың бастапқы құрылымы ретінде әрбір бет элементтерінің өрескел контурын орнатыңыз. Әдетте гистограмманы есептеу және әр түрлі масштабтағы нөлдік жарықтылықты анықтау үшін масштабты кеңістіктік сүзгі, ал объектінің контурының орнын шамамен анықтау үшін контурды бағалау әдісі қолданылады. Жалғыз ерекшелік - бұл тұлғаның нақты контурына қарағанда кішігірім контуры.

Дөрекі контур алынғаннан кейін әрбір бет элементтерінде физикалық контур табады. Көздің немесе ауыздың контурын тек жергілікті жиектер жиынтықтары кәдімгі жиек детекторлары арқылы дәл анықтай алмайды. Мәселе мынада: әдеттегі шеттік детекторлар жергілікті ақпаратты объектінің бүкіләлемдік контурына синтездеуге мүмкіндік бермейді. Сондықтан көз детекторының дизайны деформацияланатын стандартты әдіске негізделген, ол күтілетін пішін туралы априорлық ақпаратпен анықталған және жаттығу процесінде қолданылатын параметрлер жиынтығымен анықталады. Бұл стандарттар икемді және олардың өлшемдерін өзгерту арқылы өлшемдері мен формаларын өзгертеді, өйткені стандарт кескінмен өзара әрекеттеседі. Стандартты сипаттау параметрлерінің алынған мәндері тұлғаның нақты элементтерін сипаттау үшін қолданылады.

Егер екі тәсілді салыстыратын болсақ: белгілердің векторына негізделген тұлғаны сәйкестендіру, олар EL геометриялық сипаттамалары болып табылады және сұр реңктегі стандарттарды салыстыру негізінде тұлғаны сәйкестендіру, сұр реңктегі стандарттарға негізделген корреляциялық-экстремалды тәсіл тиімдірек жұмыс істейтіндігін көруге болады. . Бұл тәсіл бет элементтерінің құрылымын априорлы түрде білуді қажет етпейді. Сонымен бірге, негізделген техника сипаттамалары айтарлықтай бет элементтерінің анықтау жылдамдығын береді, мамандандырылған бағдарламалық жасақтама мен аппараттық құралды және үлкен жадыны қажет етпейді.

1.3 Адамның келбетін анықтау қазіргі таңда және оның перспективалары

Қазір көптеген салалар адамның келбетін анықтау технологиясын қолдануда үлкен үміт күтуде. Бетті биометриялық идентификациялау - бүкіл әлемдегі ақпараттық қауіпсіздіктің ең маңызды құралдарының бірі. Бетті анықтау қазіргі уақытта әсіресе соңғы бірнеше жыл ішінде смартфондарда және корпоративті қауіпсіздікте қолданылған дауыстық аутентификация немесе саусақ ізін сканерлеу сияқты биометриялық технологиялар қатарында белсенді қолдануда.

Бет идентификациясы қолданылатын ең танымал салалардың бірі - телеком. Соған мысал - телефонның құлпын ашу үшін адамның келбетін анықтау алгоритмын қолданатын «Apple» компаниясының «Face ID» технологиясы. Құрылғыларда кіріктірілген машиналық оқыту алгоритмі бар, ол әр уақытта қолданушының бетін анықтап, содан кейін оны мимика немесе аксессуарларды киіп тұрған кезде де тануға үйренеді. «Android Q» амалдық жүйесінде жарияланған ұқсас мүмкіндікті қосқанда - жақын арада 3D моделін қолдана отырып, тұлғаны анықтауға қолдау көрсетілетін болады.

Банк саласында технология жедел дамып келеді. Өткен жылдың соңында Сбербанк банкоматтарда тұлғаны анықтау технологиясын сынақтан өткізе бастады. Сонымен қатар, 2018 жылы Бірыңғай биометриялық мәліметтер базасы іске қосылды, қазірдің өзінде 150-ден астам банк өз клиенттерінен биометриялық ақпарат жинауға кірісті. Биометрияны қолдану банк клиенттеріне қаржылық қызметтерді банкоматтар арқылы ғана емес, сонымен қатар мобильді қосымшаларда алу үшін қашықтықтан сәйкестендіру мүмкіндігін ұсынады.

Бет-әлпетті сәйкестендіру технологиясы қоғамдық қауіпсіздік жүйесі үшін де маңызды қадам болып табылады. 2019 жылы қылмыскерлерді іздеу үшін Мәскеу көшелерінде жалпы қала бойынша тұлғаны анықтау жүйесі қолданылады.

Биометрика корпоративті ортада дами бастағанын айта кеткен жөн, бірақ іс жүзінде ұйымдарда осындай технологиялардың бірнеше ірі енгізілімдері бар. Сонымен қатар, киберқауіпсіздікті бұзу көбінесе сыртқы қауіп-қатерлерге байланысты емес, сонымен қатар адами факторларға байланысты анықталады. Мысалы, қызметкер жолдаманы ұмытып, әріптесінің жолдамасын пайдаланады, оны жоғалтады немесе кеңсе аумағына достарын әкеледі. Корпоративті қауіпсіздікте тұлғаны сәйкестендіру жүйесін қолдану қызметкерлерге корпоративті жүйеге кіруге, компанияның айналасында кімнің нақты қозғалатынын қадағалауға және осы адамның қызметкер екенін немесе болмауын анықтауға және т.б. көптеген мүмкіндіктер береді.

Адамның бетін сәйкестендірудің биометриялық жүйесі перспективалы аутентификация механизмі болып табылады, ол пайдаланушы тұрғысынан өте ыңғайлы - мобильді құрылғыларда ПИН-кодтар мен парольдерді бірнеше рет енгізудің қажеті жоқ. Сондай-ақ, ұмытылған пароль сияқты келеңсіздік жоғалады, егер сіз құрылғыңызды немесе корпоративті жабдықты жоғалтсаңыз, дәстүрлі бұзу әдістерінің қаупі азаяды. Алайда, егер биометрияны алдыңғы қорғаныс әдістеріне балама ретінде пайдалану жоспарланып отыр дегенді

білдіретін болсақ, оның жұмысының «жалғыз» тәуекелдері өте көп екенін ұмытпаған жөн.

Жақыннан көрінген мысалдардың бірі - Forbes журналисінің эксперименті, ол смартфондардағы биометрия жүйесінің сенімсіздігін көрсетті. Ол басының 3D көшірмесіне тапсырыс беріп, оларды LG G7 ThinQ, Samsung S9, Samsung Note 8, OnePlus 6 және iPhone X-де сынап көрді. Сылақ көшірмесі iPhone X-тен басқа смартфондардың бәрін бұғаттаумен жеңе алды.

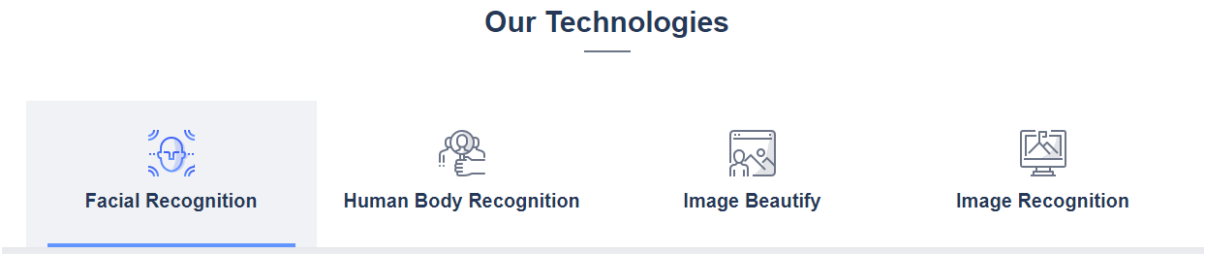
Сонымен қатар, егер биометриялық мәліметтер дискіде немесе «бұлтта» сақталса, хакерлік шабуыл ақпараттың таралуына әкеліп соқтыруы мүмкін, бірақ парольді өзгерту мүмкін емес. Биометриялық ақпараттар хакерлердің қолына түссе де, сол пароль немесе ПИН-кодқа қарағанда, олар өздерінің көзін, бетін немесе саусақ іздерін өзгерте алмайды. Қауіпсіздік жүйесі үшін тек қана биометриялық жүйені қолданы, толықтай қауіпсіздікті қамтамасыз ете алмайды.

2 ЖЕКЕ ТҰЛҒАНЫ ТАНУҒА АРНАЛҒАН ВЕБ-ҚОСЫМШАНЫ ӘЗІРЛЕУ

2.1 FacePlusPlus сервисінің тиімділігі

Адамның келбетін анықтау веб-қосымшасын әзірлеу үшін әртүрлі инструменттер қолдануға болады. Көбіне, танымал OpenCV кітапханасын қолданады, бірақ мен MEGVII Қытай компаниясының тегін Face++ (FacePlusPlus) сервисын таңдадым. Себебі олардың сервистерінде көптеген инструменттер бар, олар(1 сурет):

– Facial Recognition
– Human Body Recognition
– Image Beautify
– Image Recognition



1 сурет - FacePlusPlus технологиялары

Facial Recognition ішінде 3 түрлі инструмент бар:

– Face Detection - адамның бет-әлпетін кескін ішінен анықтау және табу, сондай-ақ жоғары дәлдіктегі бетті шектейтін қораптарды қайтарады. Face^{++} сонымен қатар болашақта пайдалану үшін әр анықталған тұлғаның метадеректерін сақтауға мүмкіндік береді;

– Face Comparing - екі тұлғаның бір адамға тиесілі екенін тексеру. Ұқсастықты бағалау үшін сенімділік шегі мен шекті мәнге ие болады;

– Face Searching - берілген беттер жиынтығынан жаңа түрге ұқсас келбеттерді табу. Face^{++} жылдам және дәл іздеу ұқсастықтардың бағасын жинау үшін сенімділік шектері мен шектерімен бірге ұқсас беттер жиынтығын береді.

Human Body Recognition инструменттері:

– Body Detection - Face^{++} адам денесін кескін ішінен анықтайды және табады, жоғары дәлдіктегі дене шектегіш қораптарын қайтарады;

– Skeleton Detection - бас, мойын, иық, шынтақ, қол, бөксе, тізе, аяқ сияқты дене компоненттерінің негізгі нүктелерін тауып, қайтарады;

– Body OutLining - бір кескін ішіндегі денелердің сұлбаларын анықтап, өзгермелі нүктелерден тұратын жолды қайтарады.

Image Beautify инструменті:

– Face Merging - шаблон кескініндегі және біріктірілген кескіндегі беттерді біріктіріп, біріктірілген кескінді қайтарады.

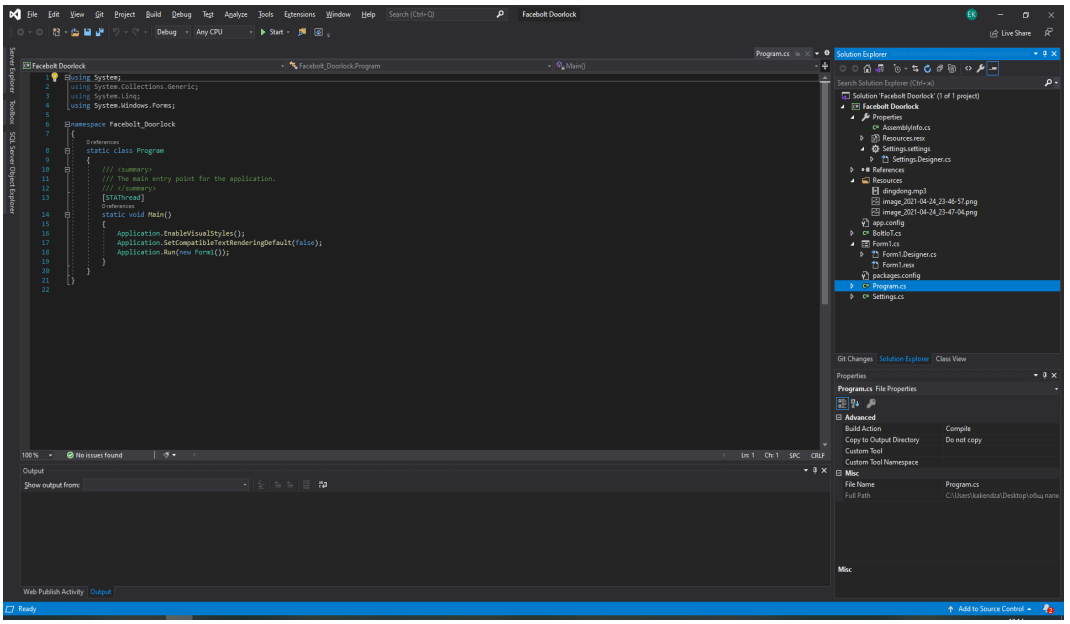
Image Recognition инструменті:

– Photo Album Clustering - альбомдағы фотосуреттердің беттерін анықтағаннан кейін, барлық фотосуреттер беттерге байланысты автоматты түрде топтастырылуы мүмкін. Содан кейін жаңа қосылған фотосуреттерді автоматты түрде кластерлеуге болады.

Осы дипломдық жұмыста Facial Recognition ішіндегі Face Detection және Face Comparing инструменттерін қолдандым.

2.2 MS Visual Studio бағдарламалау ортасы

Microsoft Visual Studio - Microsoft корпорациясының өнімі, бағдарламалық қамсыздандыру және басқа құралдарды қолдану бойынша интегрирациялық инструменталды ортасы. Бұл ортада IntelliSense қолдауы бар және қарапайым кодты қайта өңдейтін бастапқы редактор бар. Кірістірілген жөндеу құралы бастапқы деңгейдегі және машина деңгейіндегі түзеткіш ретінде жұмыс істей алады. Қалған кіріктірілген құралдарға қосымшаның GUI құруды жеңілдетуге арналған форма редакторы, веб-редактор, класс дизайнері және мәліметтер қорының схемасын құрастырушы кіреді. Visual Studio функционалдылықты барлық деңгейлерде кеңейту үшін үшінші тарап қондырмаларын (плагиндерін) жасауға және қосуға мүмкіндік береді, соның ішінде бастапқы код нұсқаларын басқару жүйелеріне қолдауды қосады (мысалы, Subversion және Visual SourceSafe сияқты), жаңа құралдар жиынтықтарын қосады (мысалы, бағдарламалық жасақтаманы әзірлеу процесінің басқа аспектілері үшін құралдар (мысалы, доменге арналған бағдарламалау тілдеріндегі визуалды дизайн коды) немесе құралдар үшін (мысалы, Team Foundation Server-мен жұмыс істеуге арналған Team Explorer клиенті). MS Visual Studio ортасын қолдану өте тиімді, себебі бұл орта IT саласында жұмыс істейтін программистерге көптеген мүмкіндіктер береді.



2 сурет - Visual Studio интерфейсі

2.3 C# бағдарламалау тілі

Қазіргі кезде C# бағдарламалау тілі IT-индустриясындағы ең қуатты, тез дамып келе жатқан және сұранысқа ие тілдердің бірі болып табылады. Қазіргі уақытта оған әр түрлі қосымшалар жазылған: шағын үстелдік бағдарламалардан бастап үлкен веб-порталдар мен күн сайын миллиондаған пайдаланушыларға қызмет көрсететін веб-қызметтер.

C# қазір жас тіл емес, cFk және бүкіл .NET платформасы ұзақ жолдан өтті. Тілдің алғашқы нұсқасы 2002 жылдың ақпанында Microsoft Visual Studio .NET шығарылымымен бірге шығарылды. Тілдің қазіргі нұсқасы - C # 9.0, ол 2020 жылдың 10 қарашасында .NET 5 шығарылымымен бірге шыққан.

C# - бұл C-ге ұқсас синтаксисі бар тіл және бұл жағынан C ++ және Java-ға жақын. Сондықтан, егер сіз осы тілдердің біреуін білсеңіз, C# -ты игеру оңайырақ болады.

C# объектіге бағытталған тіл және осыған байланысты Java және C ++ тілдерінен көп нәрсе алды. Мысалы, C# полиморфизмді, мұрагерлікті, оператордың шамадан тыс жүктелуін, статикалық теруді қолдайды. Нысанға бағытталған тәсіл үлкен, бірақ сонымен бірге икемді, масштабталатын және кеңейтілетін қосымшаларды құру мәселелерін шешуге мүмкіндік береді. C# қазіргі таңда дамуын ары қарай жалғастыруда, және әрбір жаңа нұсқада лямбда, динамикалық байланыстыру, асинхронды әдістер және т.б. сияқты қызықты функционалдары пайда болды.

```
using System;
    class Program
    {
        static void Main()
        {
            Console.WriteLine("Сәлем, әлем!!!");
        }
    }
```

3 сурет - C# тілінде қысқаша программа

Осындай қысқаша кодпен, консольға «Сәлем, әлем!!!» дегем хабарлама шығара аламыз.

C# бағдарламалау тілін қолданғанда, көбінесе .NET платформалық технологиялары еске түсіп жатады (Windows Forms, WPF, ASP.NET, Xamarin). Ал керісінше .NET десе C# еске түсіп жатады. Бұл ұғымдар бір-бірімен байланысты болғанымен оларды салыстыру дұрыс емес деп ойлаймын. C # тілі арнайы .NET шеңберімен жұмыс жасау үшін жасалған, бірақ .NET тұжырымдамасының өзі біршама кеңірек ауқымды қарастырады. .NET

платформасы Майкрософт жасаған ең жақсы платформалардың бірі және бірегейі деуге болады. .NET платформасы қосымшаларды құрудың қуатты негізін ұсынады:

– Бірнеше тілге қолдау. Платформа жалпы тілдік жұмыс уақытына (CLR) негізделген, соның арқасында .NET бірнеше тілді қолдайды: C # -пен қатар VB.NET, C ++, F #, сонымен қатар басқа тілдердің әртүрлі диалектілерін қамтиды .NET-ке байланысты, мысалы, Delphi. NET. Компиляция кезінде осы тілдердің кез-келгенінде код .NET платформасы үшін ассемблер тілі болып табылатын жалпы орта тілде (CIL) құрастырылады. Сондықтан, белгілі бір жағдайларда біз әртүрлі тілдерде бір қосымшаның бөлек модульдерін жасай аламыз;

– Кросс-платформа. .NET портативті (кейбір шектеулермен) платформа болып табылады. Мысалы, қазіргі уақытта платформаның соңғы нұсқасы - .NET 5 қазіргі заманғы Windows, MacOS, Linux ОЖ-де қолдау табады. .NET платформасында әр түрлі технологияларды қолдана отырып, C # бағдарламалау тілінде Windows, MacOS, Linux, Android, iOS, Tizen платформаларының көптеген қосымшаларын жасауға болады;

– Күшті сынып кітапханасы. .NET барлық қолдау көрсетілетін тілдер үшін бірыңғай класс кітапханасын ұсынады. Біз C#-қа қандай да бір приложение жазғымыз келеді - мәтіндік редактор, чат бөлмесі немесе күрделі веб-сайт. Басқаша түрде .NET сынып кітапханасын қолдана аламыз;

– Технологиялардың әртүрлілігі. Жалпы тілдік жұмыс уақыты (CLR) және негізгі класс кітапханасы әзірлеушілер белгілі бір қосымшаларды құру үшін қолдана алатын технологиялардың жиынтығы үшін негіз болып табылады. Мысалы, ADO.NET және Entity Framework Core осы технология стегінде мәліметтер базасымен жұмыс істеуге арналған. Бай интерфейсі бар графикалық қосымшаларды құру үшін - WPF және UWP технологиялары, қарапайым графикалық қосымшалар құру үшін - Windows Forms қолданылады. Мобильді қосымшаны дамыту үшін - Xamarin. Веб-сайттар мен веб-қосымшаларды құру үшін - ASP.NET және т.б. қолдануға болады;

– Өнімділік. Бірқатар сынақтарда .NET 5 веб-қосымшалары көптеген санаттар бойынша басқа технологиялармен құрастырылған веб-қосымшалардан озады..NET 5 қосымшалары негізінен жоғары өнімділігімен ерекшеленеді.

Сонымен қатар, C# тілінің және .NET құрылымының ерекшелігі - қоқысты автоматты түрде жинау. Бұл дегеніміз, көп жағдайда C++ тілінен басты айырмашылығы жадты босату туралы алаңдамауға болады. Жоғарыда аталған жалпы тілдік жұмыс уақыты (CLR) қоқыс жинағышты шақырады және жадыны тазартады.

C# бағдарламалау тілі осындай веб-қосымшаларды құруда ең ыңғайлы тілдердің бірі болып табылады.

2.4 .NET Windows Forms ортасы

.NET платформасын пайдаланып графикалық интерфейстерді құру үшін

әртүрлі технологиялар қолданылады. Олар, Windows Forms, WPF, Windows дүкеніне арналған қосымшалар (Windows 8 / 8.1 / 10 үшін). Алайда, ең қарапайым және ыңғайлы платформа – Windows Forms немесе формалар. Бұл нұсқаулық WinForms технологиясының көмегімен графикалық интерфейстерді құру принциптерін және негізгі басқару элементтері қалай жұмыс істейтінін түсінуге бағытталған.

Windows Forms - бұл қолданбаның графикалық интерфейсін қамтамасыз ететін және Microsoft .NET Framework құрамына кіретін қолданбалы бағдарламалау интерфейсі (API). Бұл интерфейс Win32 интерфейсінің элементтеріне қол жетімділікті басқарылатын кодқа орау арқылы жеңілдетеді. Сонымен қатар, басқарылатын код - Windows Forms үшін API енгізетін кластар даму тіліне тәуелді емес. Яғни, программист C #, C ++ және VB.Net, J #, т.б программалау тілдерінде жазу кезінде Windows Forms-ты бірдей қолдана алады.

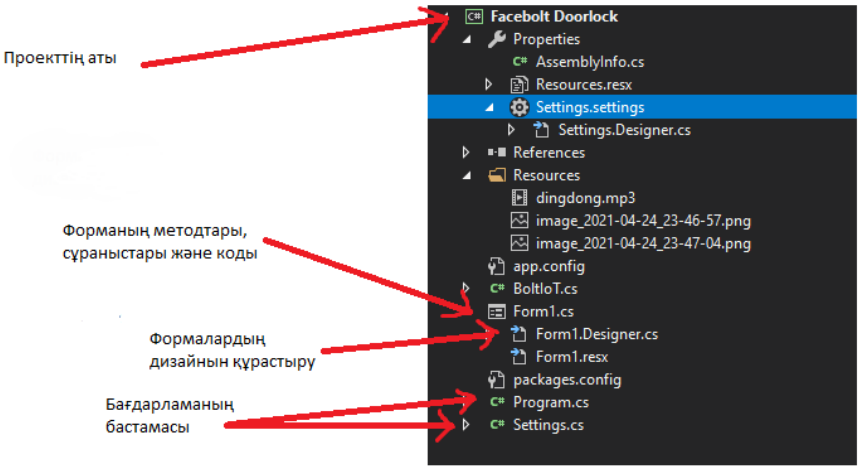
Бір жағынан, Windows Forms бастапқыда C ++ тілінде жазылған ескі және күрделі MFC кітапханасын ауыстыру ретінде қарастырылады. Екінші жағынан, WF MVC-мен салыстыруға болатын парадигманы ұсынбайды. Бұл жағдайды түзету және осы функцияны жүзеге асыру үшін Windows Forms-та үшінші тарап кітапханалары бар. Осы типтегі жиі пайдаланылатын кітапханалардың бірі - Microsoft корпорациясының арнайы мысалдары мен нұсқаулықтар тобымен тегін жүктеу үшін шығарылған қолданушы интерфейсінің қолданбалы блогы. Бұл кітапханада оқуды жеделдетуге арналған бастапқы кодтар мен оқулықтар бар.

Ішкі .NET Framework, Windows Forms жүйесі System.Windows.Forms кеңістігінде жүзеге асырылады.

Қосымшаның пайда болуы бізге ең алдымен формалар арқылы көрінеді. Пішіндер негізгі құрылыс материалы болып табылады. Олар әртүрлі басқару элементтеріне арналған контейнер ұсынады. Оқиға механизмі форма элементтеріне пайдаланушының енгізуіне жауап беруге, сөйтіп қолданушымен өзара әрекеттесуге мүмкіндік береді.

Жобаны графикалық редакторда Visual Studio-да ашқан кезде біз форманың визуалды бөлігін - қосымшаны іске қосқаннан кейін көретін және басқару панелінен элементтерді жылжытатын бөлігін көре аламыз. Бірақ іс жүзінде форма әдістерден, қасиеттерден, оқиғалардан және басқалардан тұратын қуатты функционалдылықты жасырады. Формалардың негізгі қасиеттерін қарастырайық.

Егер қосымшаны іске қосатын болсақ, бізге бір бос форма ұсынылады. Алайда бос формасы бар осындай қарапайым жобаның өзінде бірнеше компоненттер бар.

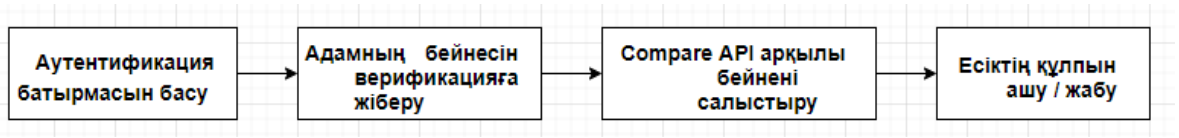


4 сурет - Жобаның компоненттері

2.5 Адамның келбетін танудың алгоритмі/схемасы

Әзірленген веб-қосымшаның жұмыс істеу схемасы.

Сырттан келген адамның келбетін сенімді адамдар базасымен салыстыру:



5 сурет - Аутентификация схемасы

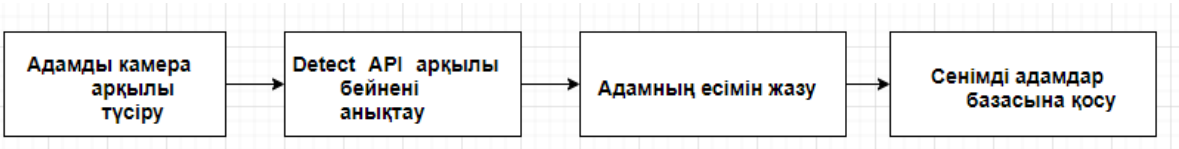
Аутентификация батырмасын басу – сыртқы есіктің қоңырау шалу батырмасын басу.

Адамның бейнесін верификацияға жіберу - камера арқылы түсірілген адамның келбетін верификацияға жіберу.

Compare API арқылы бейнені салыстыру - Compare API арқылы түсірілген бейнені локалды базадағы сенімді адамдардың келбетімен салыстыру.

Есіктің құлпын ашу / жабу – Compare API арқылы салыстырудан кейін келген жауап бойынша есіктің құлпын ашуға рұқсат беру / бермеу.

Адамның келбетін және атын сенімді адамдар базасына қосу:



6 сурет - Адамды сенімді базаға қосу схемасы

Адамды камера арқылы түсіру – сыртта тұрған камера немесе арнайы камера арқылы адамның келбетін сүретке түсіру.

Detect API арқылы бейнені анықтау – түсірілген бейненің ішінен адамның келбетін анықтап, сол бойынша жауап қайтару.

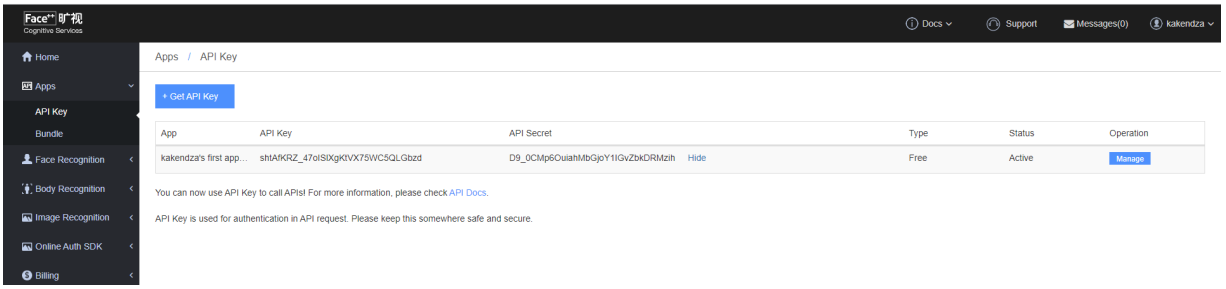
Адамның есімін жазу – Detect API-дің жауабы бойынша адамның есімін жазып алу немесе басынан суретке түсіру.

Сенімді адамдар базасына қосу / өшіру – адамның келбетін арнайы батырма арқылы қосу немесе өшіру.

3 Веб-қосымшаны әзірлеу

3.1 Face++ сервисінің сұраныстары

Face++ сервисін қолдану үшін алдымен солардың сайты арқылы регистрация өтіп, өзіміздің жеке кабинетінде арнайы API Key және API Secret деген кілттерімізді жазып аламыз. Бұл кілттер сервистің API инструменттерін қолдануға рұқсат ету үшін арналады.



7 сурет - API кілттерін алу

Кодымызды арнайы өзгертуге болмайтын айнымалыға сақтап аламыз.

```
private readonly string FPP_API_KEY = "shtAfKRZ_47oISlXgKtVX75WC5QLGbzd";
private readonly string FPP_API_SECRET = "D9_0CMp6OuiahMbGjoY1IGvZbkDRMzih";
```

8 сурет - API кілттерін жобаға қосу

Detect API үшін сұраныс сілтемесі - https://api-us.faceplusplus.com/facepp/v3/detect

Compare API үшін сұраныс сілтемесі - https://api-us.faceplusplus.com/facepp/v3/compare

Суреттер base64 кодтау жүйесі арқылы кодталып жіберіледі.

Detect API жүйесіне сұраныс жасау үлгісі:

Сұраныс параметрлері :

api_key	"shtAfKRZ_47oISlXgKtVX75WC5QLGbzd"	String
api_secret	"D9_0CMp6OuiahMbGjoY1IGvZbkDRMzih"	String
image_base64	Кодталған сурет	String

API ден қайтып келетін жауап:

request_id	Сұраныс ID-ы	String
faces	Анықталған бейнелердің массивы	Array

Compare API жүйесіне сұраныс жасау үлгісі:

Сұраныс параметрлері:

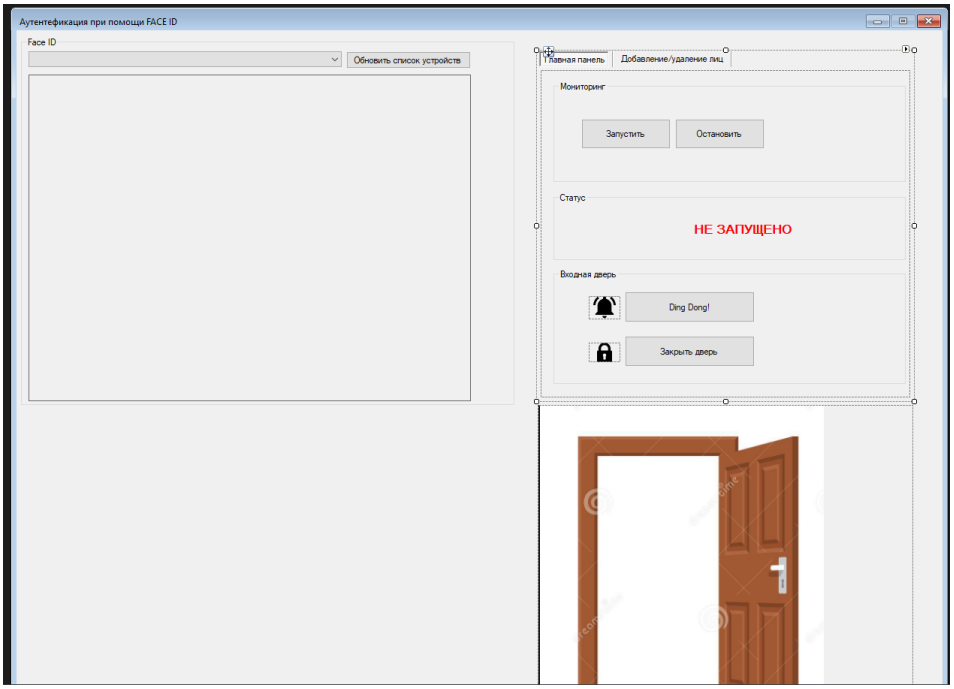
api_key	"shtAfKRZ_47oISlXgKtVX75WC5QLGbzd"	String
api_secret	"D9_0CMp6OuiahMbGjoY1IGvZbkDRMzih"	String
image_base64_1	Кодталған 1-ші сурет	String
image_base64_2	Кодталған 2-ші сурет	String

API ден қайтып келетін жауап:

request_id	Сұраныс ID-ы	String
confidence	Салыстырылған бейнелердің ұқсастық пайызы	Float

3.2 WindowsForms формаларын құру

Жүйенің интерфейсін құру үшін WindowsForms ортасының Label, PictureBox, Button, TabControl, DropDownList, DataGridView инструменттерін форманы құру үшін қолдандым. Жүйенің интерфейсі бір терезеден тұратын болады.



9 сурет - Жүйенің интерфейсі

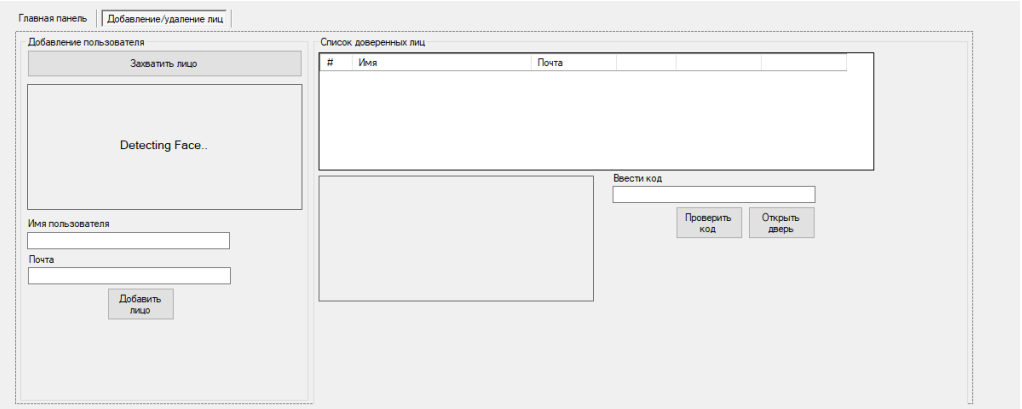
Face ID – жүйеге қосылып тұрған веб-камералар тізімі. Камераны таңдағаннан кейін, соның астында тұрған терезеден камераның түсіріп жатқанын бақылауға болады.

Главная панель бетіндегі «Мониторинг» бөлімінде тұрған «Запустить» батырмасы арқылы жүйені іске қосамыз, «Остановить» батырмасы арқылы жүйені тоқтатамыз.

Статус бөлімінде жүйенің қосылып не сөніп тұрғанын және

верификацияның мәртебесін бақылаймыз.

Входная дверь бөлімі – сыртқы есіктің виртуаланған түрі. «Ding Dong!» батырмасы арқылы қоңырау шаламыз және сол уақытта Compare API сервисі арқылы верификациядан өтеміз. «Закрыть дверь» батырмасы арқылы есіктің құлпын жабамыз.

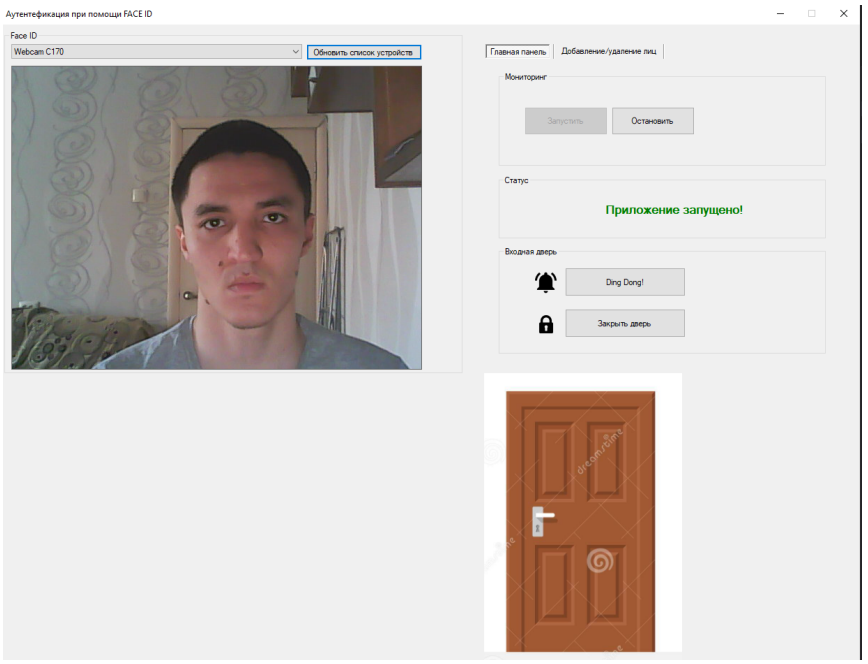


10 сурет - Жүйенің интерфейсі

Добавление/удаление лиц беті арқылы жүйемізге сенімді адамдардың келбетін қосамыз/өшіреміз. «Захватить лицо» батырмасы арқылы адамды сүретке түсіріп, Detect API сервисіне жіберіп, сервис қайтарған адам келбетінің бейнесін астындағы терезеде көрсетеміз. «Имя пользователя» терезесінде адамның атын жазып, оны локалды базада сақтап аламыз.

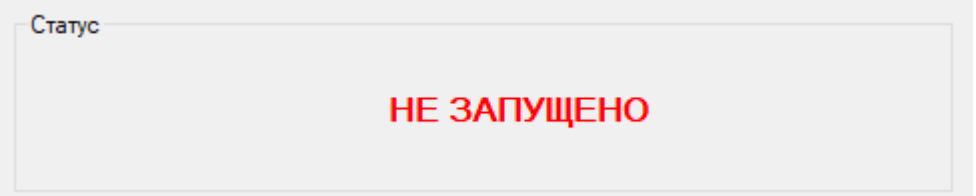
Доверенные лица кестесінде жүйеге тіркелген сенімді адамдар тізімін көруге болады, қалауымыз бойынша адамды сол тізімнен өшіре аламыз.

3.3 Веб-қосымшаның көрсетілімі

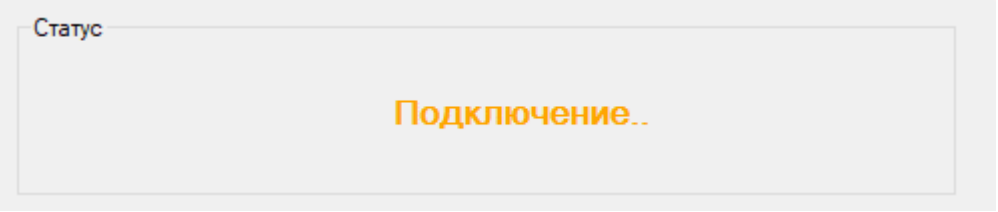


11 сурет - Жүйені іске қосқандағы интерфейсі

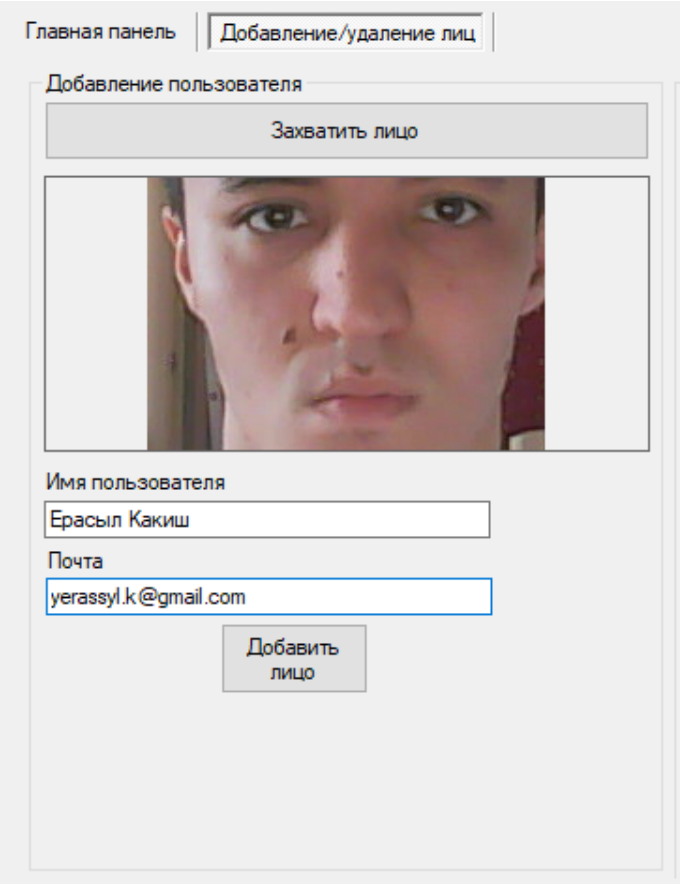
11 сурет бойынша «Статус» бөлімінде жүйенің іске қосылып тұрған мәртебесін көре аламыз.



12 сурет - Жүйе іске қосылмай тұрғандағы мәртебесі

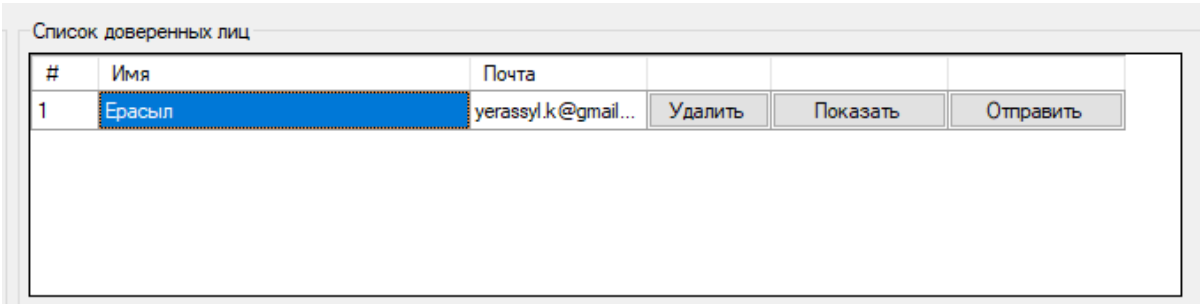


13 сурет - Жүйені іске қосқандағы мәртебесі



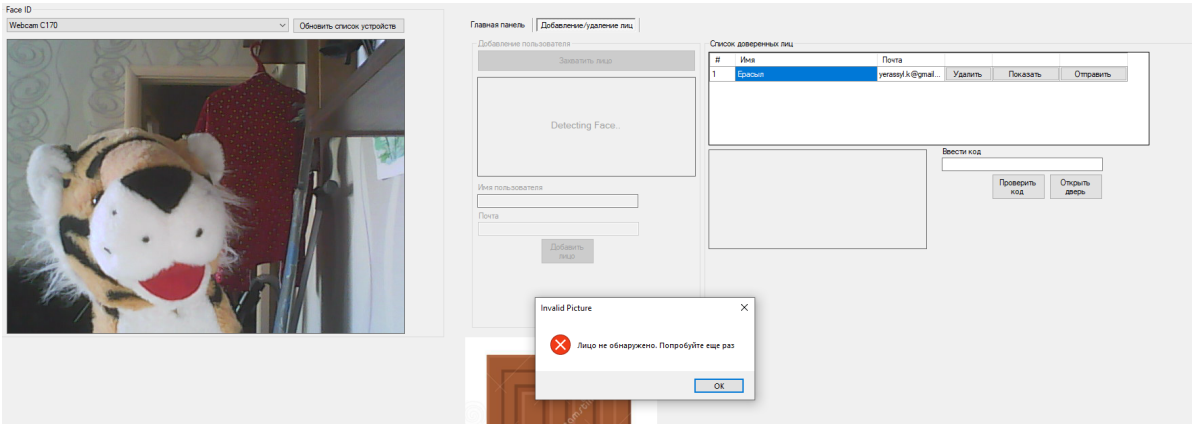
14 сурет - Жүйеге сенімді адамды қосу

«Захватить лицо» батырмасын басқан кезде, адамды суретке түсіріп, Detect API сервисіне сұраныс жасап, суреттегі адамның бейнесін терезеден көрсетеміз. Сол адамның атын және жеке почтасын жазып, «Добавить лицо» батырмасы арқылы тізімге қосып қоямыз.



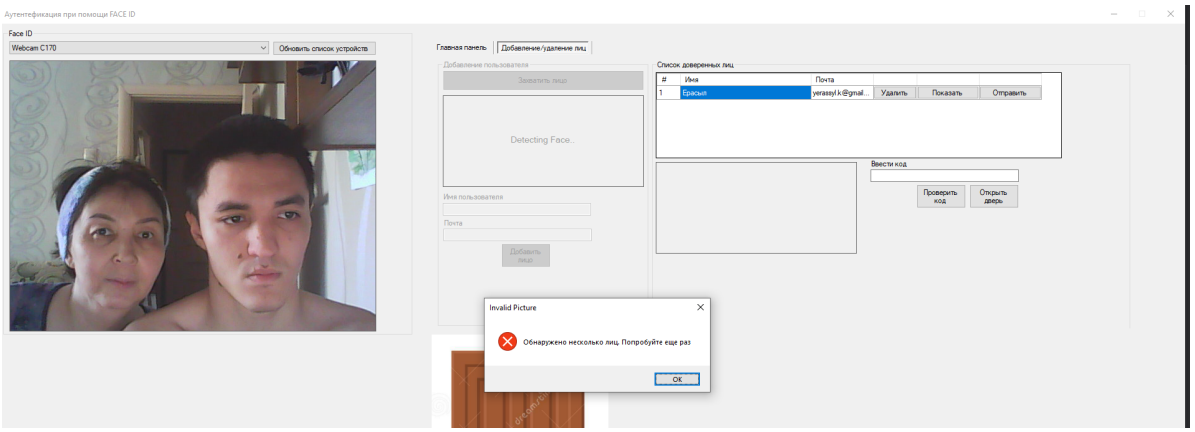
15 сурет – Сенімді адамдар тізімі

Егер, түсірілген суретте адамның келбеті табылмаса, мынадай ұяшық шығады:



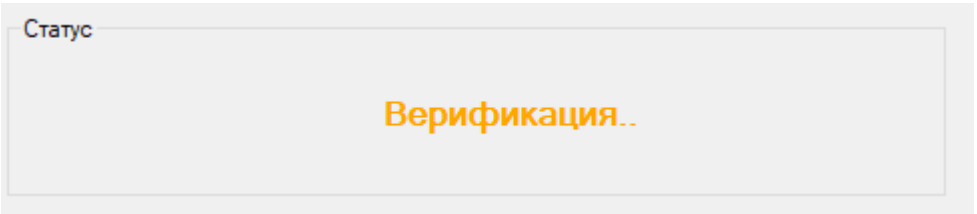
16 сурет - Адам келбеті табылмағандағы жүйенің жауабы

Егер, түсірілген суретте адамның саны 1 ден артық болса, мынадай ұяшық шығады:



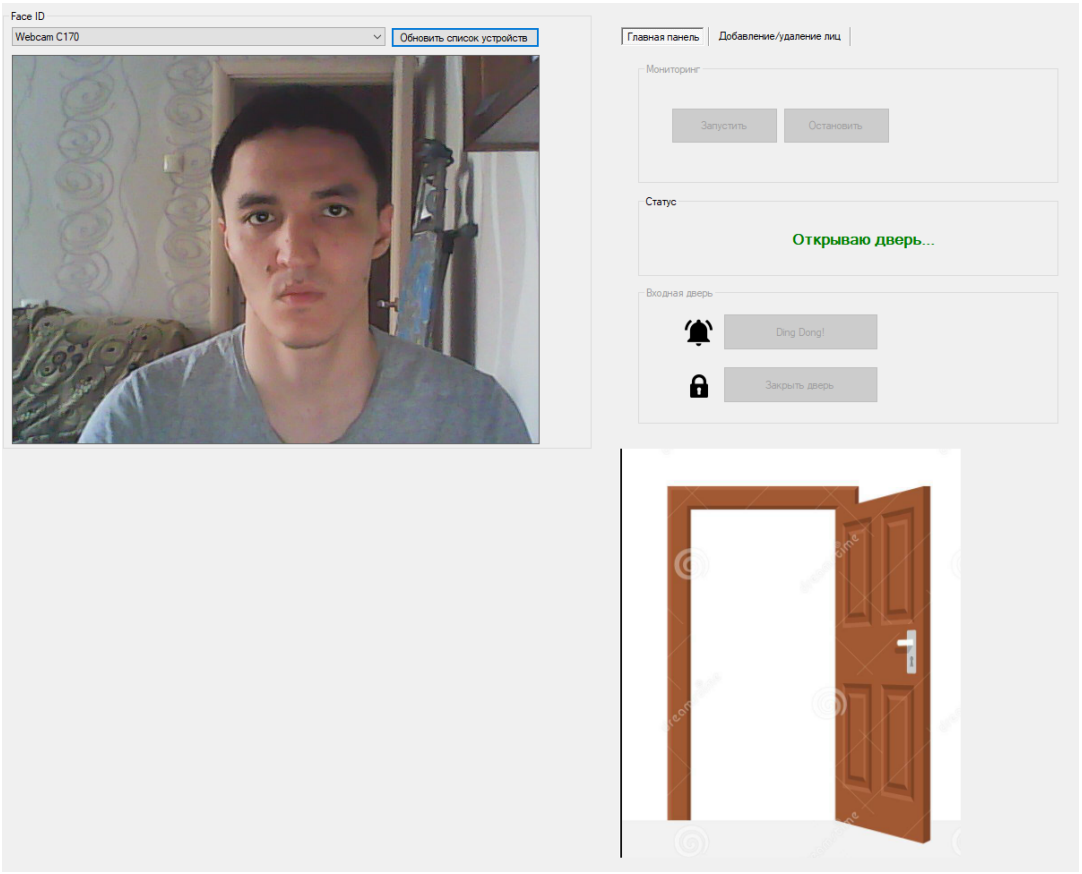
17 сурет - Адам саны 1-ден артық болғандағы жүйенің жауабы

«Ding Dong!» батырмасын басқан кезде жүйенің мәртебесі «Верификация» болып өзгереді, сол уақытта Compare API сервисіне сұраныс жасаймыз.



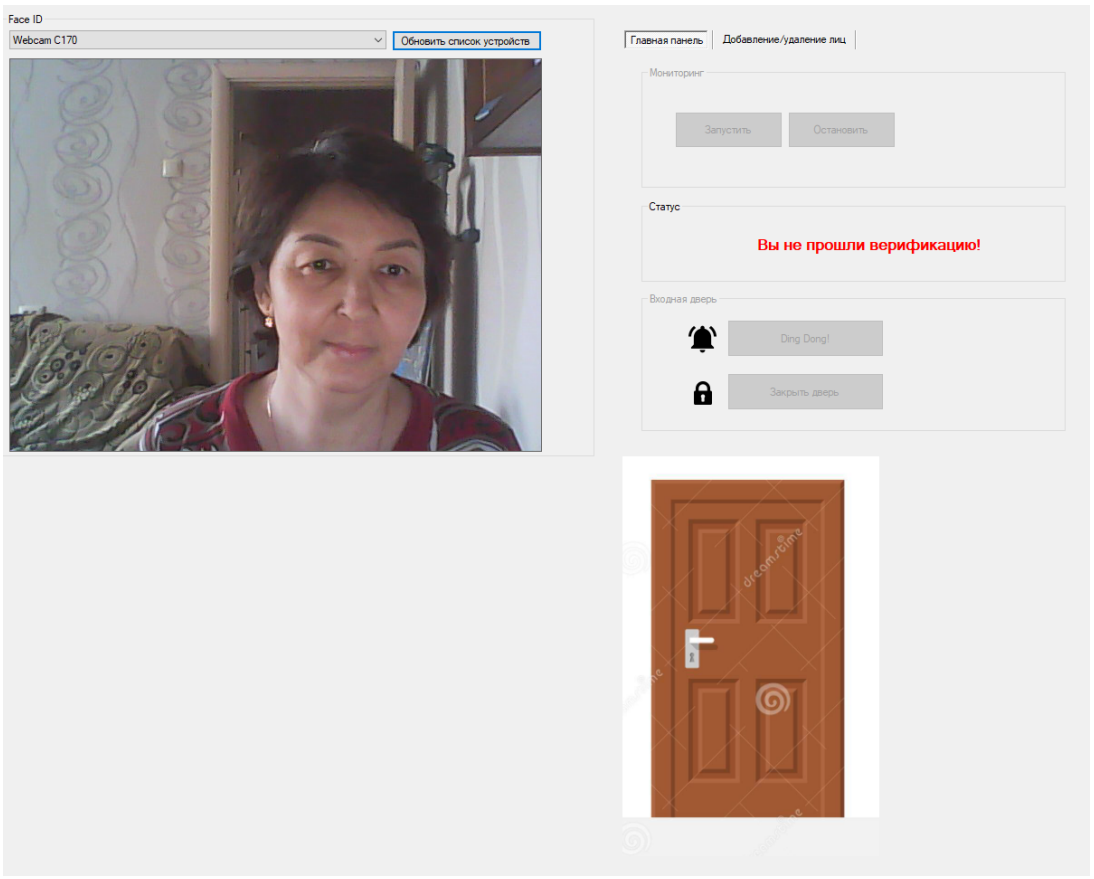
18 сурет - Жүйенің мәртебесі

Сервистің қайтарған жауабы бойынша, түсірілген адамның келбеті сенімді адамдар тізіміне сәйкес келсе, жүйенің мәртебесі «Открываю дверь» деп өзгереді және виртуалды түрдегі сыртқы есік ашылады(19 сурет).

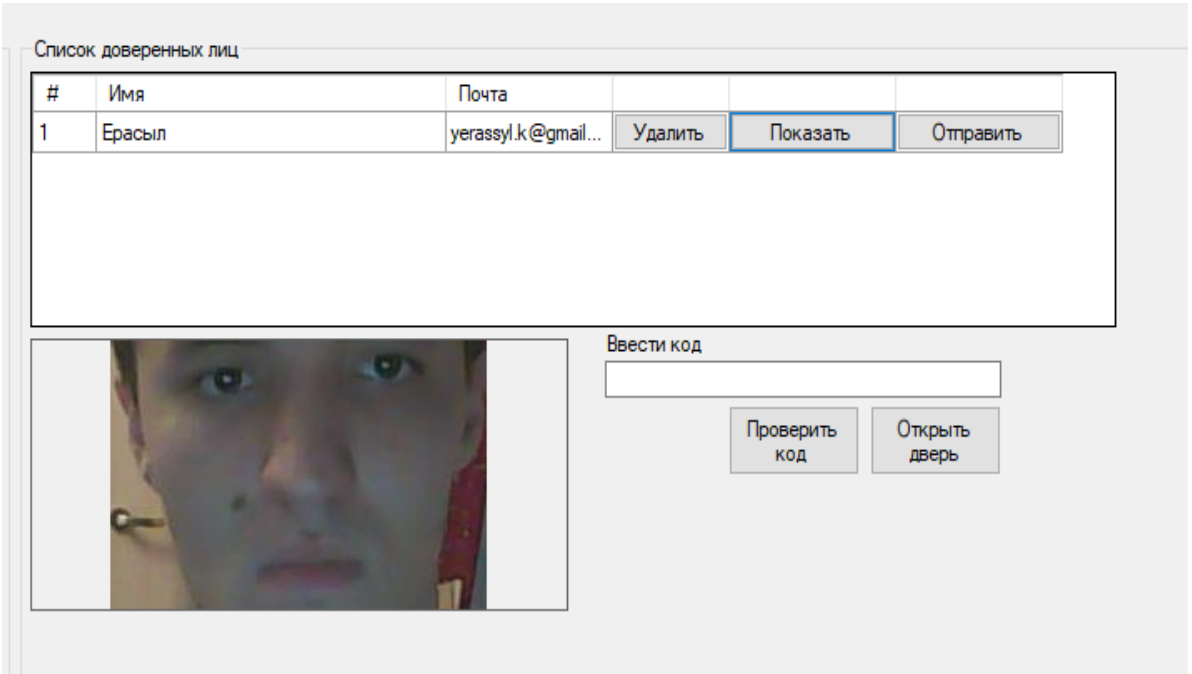


19 сурет – Сыртқы есіктің ашылуы

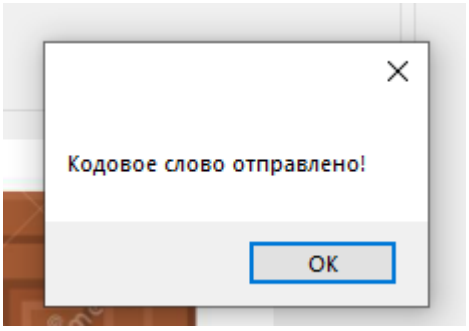
Егер түсірілген адамның келбеті сәйкес келмесе, жүйенің мәртебесі «Вы не прошли верификацию» деп өзгереді және виртуалды сыртқы есік жабылады немесе сол күйі жабық тұрады(20 сурет).



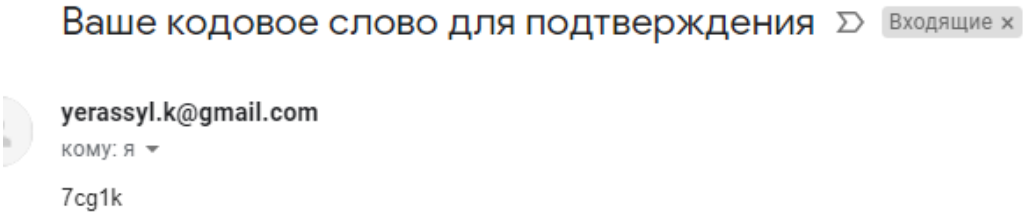
20 сурет - Верификация өтпеген жағдайдағы жүйенің мәртебесі/жауабы



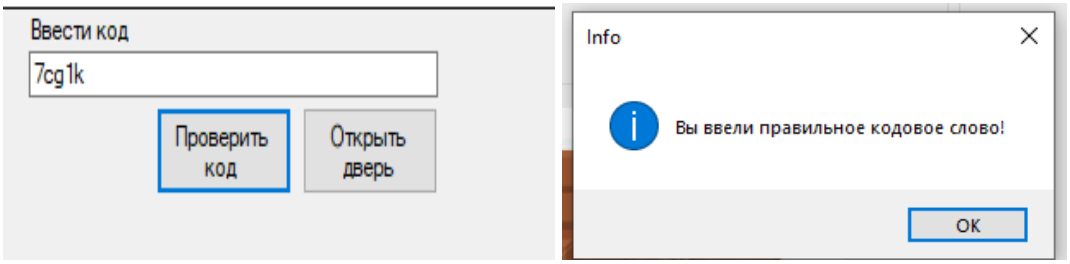
21 сурет - Деректер қорынан сенімді адамның келбетін көру



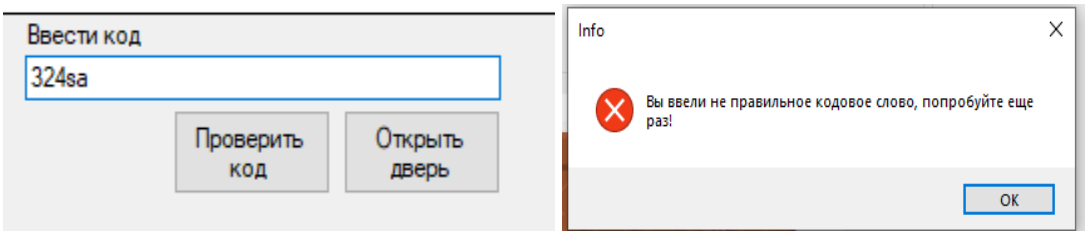
22 сурет - Сенімді адамға құпия сөз жіберу



23 сурет - Құпия сөздің электронды поштаға келуі



24 сурет - Құпия сөз сәйкес келгендегі жүйенің жауабы



25 сурет - Құпия сөз сәйкес келмегендегі жүйенің жауабы

Егер адамды анықтау жүйесі сол адамды анықтай алмай жатса, сол жақта отырған күзетші деректер қоры арқылы адамның келбетінің суретін қарап немесе сол адамның жеке почтасына құпия сөзді жіберу арқылы адамға есікті ашып бере алады.

ҚОРЫТЫНДЫ

Бұл дипломдық жұмыстың басты мақсаты – адамның келбетін анықтау тәсілдерін зерттеу, оны веб-қосымша ретінде жүйеге асыру мүмкіндігін беретін құралдарын талдау және қолдану. Дипломдық жұмыстың басты мақсатын орындайтын веб-қосымша әзірленді, бетті анықтау алгоритмі ретінде Face++ сервисі қолданылды. Веб-қосымша MS Visual Studio арнайы бағдарламалық ортада, C# бағдарламау тілі арқылы, .NET Windows Forms ортасында әзірленді.

Жүйе арқылы, үйіміздің, жеке меншік шкафтың, сейф сияқты заттар сақтау қоймаларының қауіпсіздігін қамтамасыз етуге болады. Биометриялық қорғаныстың басты артықшылығы – биометриялық деректерді көшіруге болмайтындығы.

Веб-қосымшаны қазіргі таңда қолдануға көптеген мүмкіндіктер қарастырылған: жаңадан салынып жатқан үйлерді осындай жүйемен қамтамасыз етуге болады. Себебі, жүйе үлкен шығындарға әкелмейді. Сыртқы есік жақтаушысының жанында кішігірім, адамның орташа бойындағы биіктікте тұратын құрылғы болады. Құрылғының функционалы:

- адамның келбетін салыстыру арқылы есікті ашу/жабу;
- адамның келбетін сенімді адамдар тізіміне қосу;
- басқа жағдайларда есікті пин-код арқылы ашу;
- ұры немесе бейтаныс адам келгенде, үй иесіне хат жіберу.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1 https://ru.wikipedia.org/

2 https://securityrussia.com/blog/face-recognition.html

3 https://www.faceplusplus.com/

4 https://visualstudio.microsoft.com/ru/

5 Troelsen, Andrew, Japikse, Philip, Pro C# 7 With .NET and .NET Core, 2017

6 Chris Sells, Windows Forms Programming in C#, 2003

7 Matthias Biehl, API Design, 2016

8 Asit Kumar Datta, Madhura Datta, Pradipta Kumar Banerjee, Face Detection and Recognition, 2015

9 Kelly A. Gates, Our Biometric Future: Facial Recognition Technology and the Culture of Surveillance, 2011

10 Himanshu Singh, Practical Machine Learning and Image Processing, 2019

А қосымшасы

Form1.Designer.cs – форманың ішіндегі қолданылатын инструменттер

```
namespace Facebolt_Doorlock
{
   partial class Form1
   {
      /// <summary>
      /// Required designer variable.
      /// </summary>
      private System.ComponentModel.IContainer components = null;

      /// <summary>
      /// Clean up any resources being used.
      /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
      protected override void Dispose(bool disposing)
      {
         if (disposing && (components != null))
         {
            components.Dispose();
         }
         base.Dispose(disposing);
      }

      #region Windows Form Designer generated code

      /// <summary>
      /// Required method for Designer support - do not modify
      /// the contents of this method with the code editor.
      /// </summary>
      private void InitializeComponent()
      {
            System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(Form1));
         this.button1 = new System.Windows.Forms.Button();
         this.Monitor = new System.Windows.Forms.PictureBox();
         this.VideoDevices = new System.Windows.Forms.ComboBox();
         this.groupBox1 = new System.Windows.Forms.GroupBox();
         this.tabPage2 = new System.Windows.Forms.TabPage();
         this.AddNewGB = new System.Windows.Forms.GroupBox();
         this.DecFacLabel = new System.Windows.Forms.Label();
         this.NameLabel = new System.Windows.Forms.Label();
```

А қосымшасының жалғасы

```
this.FaceNameTextBox = new System.Windows.Forms.TextBox();
        this.AddFaceButton = new System.Windows.Forms.Button();
        this.SnapFaceButton = new System.Windows.Forms.Button();
        this.PreviewBox = new System.Windows.Forms.PictureBox();
        this.groupBox2 = new System.Windows.Forms.GroupBox();
        this.TrustedFacesList = new System.Windows.Forms.DataGridView();
        this.SlNo = new System.Windows.Forms.DataGridViewTextBoxColumn();
                                                    this.PersonName = new
System.Windows.Forms.DataGridViewTextBoxColumn();
                                                    this.RemoveOption = new
System.Windows.Forms.DataGridViewButtonColumn();
        this.tabPage1 = new System.Windows.Forms.TabPage();
        this.FrontDoorGB = new System.Windows.Forms.GroupBox();
        this.pictureBox1 = new System.Windows.Forms.PictureBox();
        this.LockDoorButton = new System.Windows.Forms.Button();
        this.pictureBox3 = new System.Windows.Forms.PictureBox();
        this.RingBell = new System.Windows.Forms.Button();
        this.groupBox4 = new System.Windows.Forms.GroupBox();
        this.StatusLabel = new System.Windows.Forms.Label();
        this.MonitorControls = new System.Windows.Forms.GroupBox();
        this.StartMonitorButton = new System.Windows.Forms.Button();
        this.StopMonitorButton = new System.Windows.Forms.Button();
        this.tabControl1 = new System.Windows.Forms.TabControl();
        this.pictureBox2 = new System.Windows.Forms.PictureBox();
        ((System.ComponentModel.ISupportInitialize)(this.Monitor)).BeginInit();
        this.groupBox1.SuspendLayout();
        this.tabPage2.SuspendLayout();
        this.AddNewGB.SuspendLayout();
        ((System.ComponentModel.ISupportInitialize)(this.PreviewBox)).BeginInit();
        this.groupBox2.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.TrustedFacesList)).BeginInit();
        this.tabPage1.SuspendLayout();
        this.FrontDoorGB.SuspendLayout();
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox3)).BeginInit();
        this.groupBox4.SuspendLayout();
        this.MonitorControls.SuspendLayout();
        this.tabControl1.SuspendLayout();
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).BeginInit();
        this.SuspendLayout();
        //
        // button1
```

А қосымшасының жалғасы

```
                                                    //
            this.button1.Location = new System.Drawing.Point(433, 19);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(166, 23);
            this.button1.TabIndex = 0;
            this.button1.Text = "Обновить список устройств";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new System.EventHandler(this.RefreshButton_Click);
            //
            // Monitor
            //
            this.Monitor.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
            this.Monitor.Location = new System.Drawing.Point(10, 50);
            this.Monitor.Name = "Monitor";
            this.Monitor.Size = new System.Drawing.Size(589, 435);
                                                                this.Monitor.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.CenterImage;
            this.Monitor.TabIndex = 2;
            this.Monitor.TabStop = false;
            //
            // VideoDevices
            //
                                                        this.VideoDevices.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.VideoDevices.FormattingEnabled = true;
            this.VideoDevices.Location = new System.Drawing.Point(10, 19);
            this.VideoDevices.Name = "VideoDevices";
            this.VideoDevices.Size = new System.Drawing.Size(417, 21);
            this.VideoDevices.TabIndex = 88;
                                        this.VideoDevices.SelectedIndexChanged += new
System.EventHandler(this.VideoDevices_SelectedIndexChanged);
            //
            // groupBox1
            //
            this.groupBox1.Controls.Add(this.Monitor);
            this.groupBox1.Controls.Add(this.button1);
            this.groupBox1.Controls.Add(this.VideoDevices);
            this.groupBox1.Location = new System.Drawing.Point(6, 9);
            this.groupBox1.Name = "groupBox1";
            this.groupBox1.Size = new System.Drawing.Size(657, 491);
            this.groupBox1.TabIndex = 89;
            this.groupBox1.TabStop = false;
            this.groupBox1.Text = "Face ID";
```

А қосымшасының жалғасы

```
//
        // tabPage2
        //
        this.tabPage2.Controls.Add(this.AddNewGB);
        this.tabPage2.Controls.Add(this.groupBox2);
        this.tabPage2.Location = new System.Drawing.Point(4, 25);
        this.tabPage2.Name = "tabPage2";
        this.tabPage2.Padding = new System.Windows.Forms.Padding(3);
        this.tabPage2.Size = new System.Drawing.Size(492, 436);
        this.tabPage2.TabIndex = 1;
        this.tabPage2.Text = "Добавление/удаление лиц";
        //
        // AddNewGB
        //
        this.AddNewGB.Controls.Add(this.DecFacLabel);
        this.AddNewGB.Controls.Add(this.NameLabel);
        this.AddNewGB.Controls.Add(this.FaceNameTextBox);
        this.AddNewGB.Controls.Add(this.AddFaceButton);
        this.AddNewGB.Controls.Add(this.SnapFaceButton);
        this.AddNewGB.Controls.Add(this.PreviewBox);
        this.AddNewGB.Location = new System.Drawing.Point(6, 6);
        this.AddNewGB.Name = "AddNewGB";
        this.AddNewGB.Size = new System.Drawing.Size(338, 259);
        this.AddNewGB.TabIndex = 2;
        this.AddNewGB.TabStop = false;
        this.AddNewGB.Text = "Добавление пользователя";
        //
        // DecFacLabel
        //
        this.DecFacLabel.AutoSize = true;
          this.DecFacLabel.Font = new System.Drawing.Font("Microsoft Sans Serif",
10F);
        this.DecFacLabel.Location = new System.Drawing.Point(114, 118);
        this.DecFacLabel.Name = "DecFacLabel";
        this.DecFacLabel.Size = new System.Drawing.Size(111, 17);
        this.DecFacLabel.TabIndex = 93;
        this.DecFacLabel.Text = "Detecting Face..";
        this.DecFacLabel.Visible = false;
        //
        // NameLabel
        //
        this.NameLabel.AutoSize = true;
        this.NameLabel.Enabled = false;
```

А қосымшасының жалғасы

```
this.NameLabel.Location = new System.Drawing.Point(6, 212);
        this.NameLabel.Name = "NameLabel";
        this.NameLabel.Size = new System.Drawing.Size(103, 13);
        this.NameLabel.TabIndex = 92;
        this.NameLabel.Text = "Имя пользователя";
        //
        // FaceNameTextBox
        //
        this.FaceNameTextBox.Enabled = false;
        this.FaceNameTextBox.Location = new System.Drawing.Point(8, 229);
        this.FaceNameTextBox.Name = "FaceNameTextBox";
        this.FaceNameTextBox.Size = new System.Drawing.Size(238, 20);
        this.FaceNameTextBox.TabIndex = 0;
        //
        // AddFaceButton
        //
        this.AddFaceButton.Enabled = false;
        this.AddFaceButton.Location = new System.Drawing.Point(252, 213);
        this.AddFaceButton.Name = "AddFaceButton";
        this.AddFaceButton.Size = new System.Drawing.Size(79, 38);
        this.AddFaceButton.TabIndex = 90;
        this.AddFaceButton.Text = "Добавить лицо";
        this.AddFaceButton.UseVisualStyleBackColor = true;
                                                this.AddFaceButton.Click     +=     new
System.EventHandler(this.AddFaceButton_Click);
        //
        // SnapFaceButton
        //
        this.SnapFaceButton.Location = new System.Drawing.Point(8, 16);
        this.SnapFaceButton.Name = "SnapFaceButton";
        this.SnapFaceButton.Size = new System.Drawing.Size(323, 32);
        this.SnapFaceButton.TabIndex = 89;
        this.SnapFaceButton.Text = "Захватить лицо";
        this.SnapFaceButton.UseVisualStyleBackColor = true;
                                                this.SnapFaceButton.Click     +=     new
System.EventHandler(this.SnapFaceButton_Click);
        //
        // PreviewBox
        //
                                                        this.PreviewBox.BorderStyle     =
System.Windows.Forms.BorderStyle.FixedSingle;
        this.PreviewBox.Location = new System.Drawing.Point(8, 56);
        this.PreviewBox.Name = "PreviewBox";
```

А қосымшасының жалғасы

```
this.PreviewBox.Size = new System.Drawing.Size(323, 147);
                                                    this.PreviewBox.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.CenterImage;
        this.PreviewBox.TabIndex = 89;
        this.PreviewBox.TabStop = false;
        //
        // groupBox2
        //
        this.groupBox2.Controls.Add(this.TrustedFacesList);
        this.groupBox2.Location = new System.Drawing.Point(6, 271);
        this.groupBox2.Name = "groupBox2";
        this.groupBox2.Size = new System.Drawing.Size(338, 163);
        this.groupBox2.TabIndex = 1;
        this.groupBox2.TabStop = false;
        this.groupBox2.Text = "Доверенные лица";
        //
        // TrustedFacesList
        //
        this.TrustedFacesList.AllowUserToAddRows = false;
                                            this.TrustedFacesList.BackgroundColor =
System.Drawing.SystemColors.Window;
                                this.TrustedFacesList.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
                                        this.TrustedFacesList.Columns.AddRange(new
System.Windows.Forms.DataGridViewColumn[] {
        this.SlNo,
        this.PersonName,
        this.RemoveOption});
        this.TrustedFacesList.Location = new System.Drawing.Point(8, 19);
        this.TrustedFacesList.Name = "TrustedFacesList";
                                        this.TrustedFacesList.RowHeadersBorderStyle =
System.Windows.Forms.DataGridViewHeaderBorderStyle.Single;
        this.TrustedFacesList.RowHeadersVisible = false;
                                    this.TrustedFacesList.RowHeadersWidthSizeMode =
System.Windows.Forms.DataGridViewRowHeadersWidthSizeMode.AutoSizeToAll
Headers;
        this.TrustedFacesList.Size = new System.Drawing.Size(323, 138);
        this.TrustedFacesList.TabIndex = 0;
        //
        // SlNo
        //
        this.SlNo.Frozen = true;
        this.SlNo.HeaderText = "#";
```

А қосымшасының жалғасы

```
this.SlNo.Name = "SlNo";
            this.SlNo.ReadOnly = true;
            this.SlNo.Width = 38;
            //
            // PersonName
            //
            this.PersonName.Frozen = true;
            this.PersonName.HeaderText = "Имя";
            this.PersonName.Name = "PersonName";
            this.PersonName.ReadOnly = true;
            this.PersonName.Width = 210;
            //
            // RemoveOption
            //
            this.RemoveOption.Frozen = true;
            this.RemoveOption.HeaderText = "";
            this.RemoveOption.Name = "RemoveOption";
            this.RemoveOption.ReadOnly = true;
            this.RemoveOption.Width = 70;
            //
            // tabPage1
            //
            this.tabPage1.Controls.Add(this.FrontDoorGB);
            this.tabPage1.Controls.Add(this.groupBox4);
            this.tabPage1.Controls.Add(this.MonitorControls);
            this.tabPage1.Location = new System.Drawing.Point(4, 25);
            this.tabPage1.Name = "tabPage1";
            this.tabPage1.Padding = new System.Windows.Forms.Padding(3);
            this.tabPage1.Size = new System.Drawing.Size(492, 436);
            this.tabPage1.TabIndex = 0;
            this.tabPage1.Text = "Главная панель";
            //
            // FrontDoorGB
            //
            this.FrontDoorGB.Controls.Add(this.pictureBox1);
            this.FrontDoorGB.Controls.Add(this.LockDoorButton);
            this.FrontDoorGB.Controls.Add(this.pictureBox3);
            this.FrontDoorGB.Controls.Add(this.RingBell);
            this.FrontDoorGB.Enabled = false;
            this.FrontDoorGB.Location = new System.Drawing.Point(17, 265);
            this.FrontDoorGB.Name = "FrontDoorGB";
            this.FrontDoorGB.Size = new System.Drawing.Size(469, 154);
            this.FrontDoorGB.TabIndex = 94;
```

А қосымшасының жалғасы

```
this.FrontDoorGB.TabStop = false;
        this.FrontDoorGB.Text = "Входная дверь";
        //
        // pictureBox1
        //
        this.pictureBox1.BackColor = System.Drawing.SystemColors.Control;
                                                        this.pictureBox1.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox1.Image")));
        this.pictureBox1.Location = new System.Drawing.Point(47, 98);
        this.pictureBox1.Name = "pictureBox1";
        this.pictureBox1.Size = new System.Drawing.Size(42, 26);
                                                    this.pictureBox1.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
        this.pictureBox1.TabIndex = 96;
        this.pictureBox1.TabStop = false;
        //
        // LockDoorButton
        //
        this.LockDoorButton.Location = new System.Drawing.Point(95, 89);
        this.LockDoorButton.Name = "LockDoorButton";
        this.LockDoorButton.Size = new System.Drawing.Size(173, 41);
        this.LockDoorButton.TabIndex = 95;
        this.LockDoorButton.Text = "Закрыть дверь";
        this.LockDoorButton.UseVisualStyleBackColor = true;
                                            this.LockDoorButton.Click += new
System.EventHandler(this.LockDoorButton_Click);
        //
        // pictureBox3
        //
        this.pictureBox3.BackColor = System.Drawing.SystemColors.Control;
                                                        this.pictureBox3.Image =
((System.Drawing.Image)(resources.GetObject("pictureBox3.Image")));
        this.pictureBox3.Location = new System.Drawing.Point(47, 36);
        this.pictureBox3.Name = "pictureBox3";
        this.pictureBox3.Size = new System.Drawing.Size(42, 31);
                                                    this.pictureBox3.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
        this.pictureBox3.TabIndex = 93;
        this.pictureBox3.TabStop = false;
                                                this.pictureBox3.Click += new
System.EventHandler(this.pictureBox3_Click);
        //
        // RingBell
```

```
        //
```

А қосымшасының жалғасы

```
this.RingBell.Location = new System.Drawing.Point(95, 30);
        this.RingBell.Name = "RingBell";
        this.RingBell.Size = new System.Drawing.Size(173, 41);
        this.RingBell.TabIndex = 89;
        this.RingBell.Text = "Ding Dong!";
        this.RingBell.UseVisualStyleBackColor = true;
        this.RingBell.Click += new System.EventHandler(this.RingBell_Click);
        //
        // groupBox4
        //
        this.groupBox4.Controls.Add(this.StatusLabel);
        this.groupBox4.Location = new System.Drawing.Point(17, 163);
        this.groupBox4.Name = "groupBox4";
        this.groupBox4.Size = new System.Drawing.Size(469, 91);
        this.groupBox4.TabIndex = 93;
        this.groupBox4.TabStop = false;
        this.groupBox4.Text = "Статус";
        //
        // StatusLabel
        //
            this.StatusLabel.Font = new System.Drawing.Font("Microsoft Sans Serif",
12F, System.Drawing.FontStyle.Bold);
        this.StatusLabel.ForeColor = System.Drawing.Color.Red;
        this.StatusLabel.Location = new System.Drawing.Point(42, 28);
        this.StatusLabel.Name = "StatusLabel";
        this.StatusLabel.Size = new System.Drawing.Size(421, 42);
        this.StatusLabel.TabIndex = 93;
        this.StatusLabel.Text = "НЕ ЗАПУЩЕНО";
                                                this.StatusLabel.TextAlign      =
System.Drawing.ContentAlignment.MiddleCenter;
        this.StatusLabel.Click += new System.EventHandler(this.StatusLabel_Click);
        //
        // MonitorControls
        //
        this.MonitorControls.Controls.Add(this.StartMonitorButton);
        this.MonitorControls.Controls.Add(this.StopMonitorButton);
        this.MonitorControls.Location = new System.Drawing.Point(17, 15);
        this.MonitorControls.Name = "MonitorControls";
        this.MonitorControls.Size = new System.Drawing.Size(469, 135);
        this.MonitorControls.TabIndex = 92;
        this.MonitorControls.TabStop = false;
        this.MonitorControls.Text = "Мониторинг";
```

```
        this.MonitorControls.Enter += new
System.EventHandler(this.MonitorControls_Enter);
        //
        // StartMonitorButton
        //
        this.StartMonitorButton.Location = new System.Drawing.Point(37, 50);
        this.StartMonitorButton.Name = "StartMonitorButton";
        this.StartMonitorButton.Size = new System.Drawing.Size(119, 40);
        this.StartMonitorButton.TabIndex = 90;
        this.StartMonitorButton.Text = "Запустить";
        this.StartMonitorButton.UseVisualStyleBackColor = true;
                                        this.StartMonitorButton.Click += new
System.EventHandler(this.StartMonitorButton_Click);
        //
        // StopMonitorButton
        //
        this.StopMonitorButton.Enabled = false;
        this.StopMonitorButton.Location = new System.Drawing.Point(162, 50);
        this.StopMonitorButton.Name = "StopMonitorButton";
        this.StopMonitorButton.Size = new System.Drawing.Size(119, 40);
        this.StopMonitorButton.TabIndex = 91;
        this.StopMonitorButton.Text = "Остановить";
        this.StopMonitorButton.UseVisualStyleBackColor = true;
                                        this.StopMonitorButton.Click += new
System.EventHandler(this.StopMonitorButton_Click);
        //
        // tabControl1
        //
                                                this.tabControl1.Appearance =
System.Windows.Forms.TabAppearance.FlatButtons;
        this.tabControl1.Controls.Add(this.tabPage1);
        this.tabControl1.Controls.Add(this.tabPage2);
        this.tabControl1.Location = new System.Drawing.Point(694, 28);
        this.tabControl1.Name = "tabControl1";
        this.tabControl1.SelectedIndex = 0;
        this.tabControl1.Size = new System.Drawing.Size(500, 465);
        this.tabControl1.TabIndex = 90;
        //
        // pictureBox2
        //
                                                    this.pictureBox2.Image =
global::Facebolt_Doorlock.Properties.Resources.image_2021_04_24_23_46_57;
        this.pictureBox2.Location = new System.Drawing.Point(694, 499);
```

```
        this.pictureBox2.Name = "pictureBox2";
this.pictureBox2.Size = new System.Drawing.Size(500, 471);
        this.pictureBox2.TabIndex = 91;
        this.pictureBox2.TabStop = false;
        //
        // Form1
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(1233, 996);
        this.Controls.Add(this.pictureBox2);
        this.Controls.Add(this.tabControl1);
        this.Controls.Add(this.groupBox1);
                                                this.FormBorderStyle =
System.Windows.Forms.FormBorderStyle.FixedDialog;
        this.MaximizeBox = false;
        this.Name = "Form1";
        this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
        this.Text = "Аутентефикация при помощи FACE ID";
                                                this.FormClosing += new
System.Windows.Forms.FormClosingEventHandler(this.Form1_FormClosing);
        this.Load += new System.EventHandler(this.Form1_Load);
        ((System.ComponentModel.ISupportInitialize)(this.Monitor)).EndInit();
        this.groupBox1.ResumeLayout(false);
        this.tabPage2.ResumeLayout(false);
        this.AddNewGB.ResumeLayout(false);
        this.AddNewGB.PerformLayout();
        ((System.ComponentModel.ISupportInitialize)(this.PreviewBox)).EndInit();
        this.groupBox2.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.TrustedFacesList)).EndInit();
        this.tabPage1.ResumeLayout(false);
        this.FrontDoorGB.ResumeLayout(false);
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox3)).EndInit();
        this.groupBox4.ResumeLayout(false);
        this.MonitorControls.ResumeLayout(false);
        this.tabControl1.ResumeLayout(false);
        ((System.ComponentModel.ISupportInitialize)(this.pictureBox2)).EndInit();
        this.ResumeLayout(false);

    }
```

```
        #endregion
```

А қосымшасының жалғасы

```
private System.Windows.Forms.Button button1;
        private System.Windows.Forms.PictureBox Monitor;
        private System.Windows.Forms.ComboBox VideoDevices;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.TabPage tabPage2;
        private System.Windows.Forms.GroupBox AddNewGB;
        private System.Windows.Forms.Label DecFacLabel;
        private System.Windows.Forms.Label NameLabel;
        private System.Windows.Forms.TextBox FaceNameTextBox;
        private System.Windows.Forms.Button AddFaceButton;
        private System.Windows.Forms.Button SnapFaceButton;
        private System.Windows.Forms.PictureBox PreviewBox;
        private System.Windows.Forms.GroupBox groupBox2;
        private System.Windows.Forms.DataGridView TrustedFacesList;
        private System.Windows.Forms.TabPage tabPage1;
        private System.Windows.Forms.GroupBox FrontDoorGB;
        private System.Windows.Forms.PictureBox pictureBox1;
        private System.Windows.Forms.Button LockDoorButton;
        private System.Windows.Forms.PictureBox pictureBox3;
        private System.Windows.Forms.Button RingBell;
        private System.Windows.Forms.GroupBox groupBox4;
        private System.Windows.Forms.Label StatusLabel;
        private System.Windows.Forms.GroupBox MonitorControls;
        private System.Windows.Forms.Button StartMonitorButton;
        private System.Windows.Forms.Button StopMonitorButton;
        private System.Windows.Forms.TabControl tabControl1;
        private System.Windows.Forms.DataGridViewTextBoxColumn SlNo;
        private System.Windows.Forms.DataGridViewTextBoxColumn PersonName;
        private System.Windows.Forms.DataGridViewButtonColumn RemoveOption;
        private System.Windows.Forms.PictureBox pictureBox2;
    }
}
```

В қосымшасы

Form1.cs – форманың барлық қолданылатын іс-шаралардың методтары

```
using System;
using System.IO;
using System.Net;
using System.Drawing;
using System.Threading;
using System.Windows.Forms;
using System.Threading.Tasks;
using System.Drawing.Imaging;
using System.Speech.Synthesis;
using System.Collections.Generic;
using System.Collections.Specialized;

using BoltIoT;
using AForge.Video;
using AForge.Video.DirectShow;
using Newtonsoft.Json.Linq;

namespace Facebolt_Doorlock
{
   public partial class Form1 : Form
   {
      //Modify this with your credentials
                                 private readonly string FPP_API_KEY =
"shtAfKRZ_47oISlXgKtVX75WC5QLGbzd";
                                 private readonly string FPP_API_SECRET =
"D9_0CMp6OuiahMbGjoY1IGvZbkDRMzih";

      Bolt myBolt;
      VideoCaptureDevice CurrentCam;
      Thread BellButtonListningThread;
      FilterInfoCollection VideoDeviceCollection;

      List<string> NameList;
      List<string> ImageDataList;

      bool SecurityRunning = false, ListenForBell;

      Bitmap lockedDoor = Properties.Resources.image_2021_04_24_23_47_04;
      Bitmap unlockedDoor = Properties.Resources.image_2021_04_24_23_46_57;

      SpeechSynthesizer synthesizer = new SpeechSynthesizer
```

В қосымшасының жалғасы

```
{
        Volume = 100,  // 0...100
        Rate = 1     // -10...10
    };

    public Form1()
    {
        InitializeComponent();
        pictureBox2.Image = lockedDoor;
          synthesizer.SpeakAsync("Добро пожаловать! Пройдите верификацию что бы войти!");
    }

      private void Form1_Load(object sender, EventArgs e) //Gets called when the Application starts
    {
          //myBolt = new Bolt(BOLT_API_KEY, BOLT_DEVICE_ID); // new Bolt instance

        // Settings will be NULL in the first run. Create new list if so.
            NameList = Properties.Settings.Default.FaceNames ?? new List<string>(); //List that holds names of trusted faces
                ImageDataList = Properties.Settings.Default.Base64ImageData ?? new List<string>(); //List that holds base64 encoded strings of trusted face images
                                                    VideoDeviceCollection = new FilterInfoCollection(FilterCategory.VideoInputDevice); //Fetches camera devices connected to the system

                                //Registering button click event for Remove button. 'TrustedFacesList_CellContentClick' will be called when it is pressed.
        TrustedFacesList.CellContentClick += TrustedFacesList_CellContentClick;

        for (int i = 0; i < NameList.Count; i++)
            TrustedFacesList.Rows.Add(i + 1, NameList[i], "Удалить");

         if (VideoDeviceCollection.Count != 0) //Populates the dropdown menu with a list of camera devices connected to the system
        {
            foreach (FilterInfo cams in VideoDeviceCollection)
                VideoDevices.Items.Add(cams.Name);

            VideoDevices.SelectedIndex = 0;
        }
```

В қосымшасының жалғасы

```
 else //No camera device detected
        {
            MessageBox.Show("Не найдены подключенные камеры. Пожалуйтса, перезапустите приложение!", "Camera Missing", MessageBoxButtons.OK, MessageBoxIcon.Error);
           Environment.Exit(0);
        }

                                                    CurrentCam = new VideoCaptureDevice(VideoDeviceCollection[VideoDevices.SelectedIndex].MonikerString);
            CurrentCam.NewFrame += new NewFrameEventHandler(NewFrame); // Start live display of feed from camera
        CurrentCam.Start();
    }

                private void TrustedFacesList_CellContentClick(object sender, DataGridViewCellEventArgs e) //Method that remocves a trusted face
    {
       var senderGrid = (DataGridView)sender;

        if (senderGrid.Columns[e.ColumnIndex] is DataGridViewButtonColumn && e.RowIndex >= 0)
        {
           //Removing face information at specified position in the list
           NameList.RemoveAt(e.RowIndex);
           ImageDataList.RemoveAt(e.RowIndex);
           senderGrid.Rows.RemoveAt(e.RowIndex);

           //Saving the the list after removal of a face
           Properties.Settings.Default.FaceNames = NameList;
           Properties.Settings.Default.Base64ImageData = ImageDataList;
           Properties.Settings.Default.Save();

           for (int i = 0; i < senderGrid.RowCount; i++)
              senderGrid.Rows[i].Cells[0].Value = i + 1;
        }
    }

      private void RefreshButton_Click(object sender, EventArgs e) //Refreshes the list of camera devices connected to the system
    {
       VideoDevices.Items.Clear();
```

В қосымшасының жалғасы

```
                VideoDeviceCollection = new FilterInfoCollection(FilterCategory.VideoInputDevice);

            if (VideoDeviceCollection.Count != 0)
            {
                foreach (FilterInfo cams in VideoDeviceCollection)
                    VideoDevices.Items.Add(cams.Name);

                VideoDevices.SelectedIndex = 0;
            }
        }

         private void VideoDevices_SelectedIndexChanged(object sender, EventArgs e) // Gets called when user selects a new camera device from the list
        {
            if (CurrentCam != null && VideoDeviceCollection.Count != 0)
            {
                CurrentCam.Stop();
                                                CurrentCam = new VideoCaptureDevice(VideoDeviceCollection[VideoDevices.SelectedIndex].MonikerString);
                CurrentCam.NewFrame += new NewFrameEventHandler(NewFrame);
                CurrentCam.Start();
            }
        }

            private void NewFrame(object sender, NewFrameEventArgs eventArgs) // Method that updates picture in PictureBox during each frame
        {
            try
            {
                Monitor.Image = (Bitmap)eventArgs.Frame.Clone();
            }
            catch { }
        }

        /// <summary>
        /// Method that compares two faces
        /// </summary>
            /// <param name="face1Base64">base664 encoded string of image of face 1</param>
            /// <param name="face2Base64">base664 encoded string of image of face 2</param>
```

В қосымшасының жалғасы

```
/// <returns>Confidence percentage that two faces are similar</returns>
        public async Task<double> VerifyFace(string face1Base64, string face2Base64)
        {
            WebClient client = new WebClient();
            byte[] response = null;

            await Task.Run(delegate
            {
                                                                    response = client.UploadValues("https://api-us.faceplusplus.com/facepp/v3/compare", new NameValueCollection()
                                    {
                                        { "api_key", FPP_API_KEY },
                                        { "api_secret", FPP_API_SECRET },
                                        { "image_base64_1", face1Base64},
                                        { "image_base64_2", face2Base64}
                                    });
            });

            string data = "";
            try
            {
                                                                    data = JObject.Parse(System.Text.Encoding.UTF8.GetString(response))["confidence"].ToString();
            }
            catch
            {
                return 0;
            }

            return double.Parse(data);
        }

         private async void SnapFaceButton_Click(object sender, EventArgs e) //Method that detects and adds new trusted face
        {
            AddNewGB.Enabled = false;
            DecFacLabel.Visible = true;

            PreviewBox.Image = null;
            Image Snap = (Bitmap)Monitor.Image.Clone();
            WebClient client = new WebClient();
```

```
byte[] response = null;
```

В қосымшасының жалғасы

```
await Task.Run(delegate
        {
                                                                response =
client.UploadValues("https://api-us.faceplusplus.com/facepp/v3/detect", new
NameValueCollection()
                {
                    { "api_key", FPP_API_KEY },
                    { "api_secret", FPP_API_SECRET},
                    { "image_base64", ImageToBase64(Snap)}
                });
        }); // Checks if the captured image contains a face

                                                        JObject data =
JObject.Parse(System.Text.Encoding.UTF8.GetString(response));
        string faces = data["faces"].ToString();
        if (faces == "[]") // No face was detected
        {
                MessageBox.Show("Лицо не обнаружено. Попробуйте еще раз",
"Invalid Picture", MessageBoxButtons.OK, MessageBoxIcon.Error);
           FaceNameTextBox.Enabled = false;
           AddFaceButton.Enabled = false;
           NameLabel.Enabled = false;
        }

        else if (faces == "[1]") //Multiple faces detected
        {
            MessageBox.Show("Обнаружено несколько лиц. Попробуйте еще раз",
"Invalid Picture", MessageBoxButtons.OK, MessageBoxIcon.Error);
           FaceNameTextBox.Enabled = false;
           AddFaceButton.Enabled = false;
           NameLabel.Enabled = false;
        }
        else
        {
           //Getting the dimentions of the detected face from the API response
                                                                int width =
int.Parse(data["faces"][0]["face_rectangle"]["width"].ToString());
                                                                int height =
int.Parse(data["faces"][0]["face_rectangle"]["height"].ToString());
           int top = int.Parse(data["faces"][0]["face_rectangle"]["top"].ToString());
           int left = int.Parse(data["faces"][0]["face_rectangle"]["left"].ToString());
```

```
            //Displaying a cropped preview of the new trusted face to be added
```

В қосымшасының жалғасы

```
 PreviewBox.Image = CropImage(Snap, new Rectangle(left - 20, top - 20, width + 20, height + 20));

            FaceNameTextBox.Enabled = true;
            AddFaceButton.Enabled = true;
            NameLabel.Enabled = true;
        }

        DecFacLabel.Visible = false;
        AddNewGB.Enabled = true;
    }

      private void AddFaceButton_Click(object sender, EventArgs e) //Method that saves a newly added trusted face
    {
        if (string.IsNullOrWhiteSpace(FaceNameTextBox.Text))
        {
                    MessageBox.Show("Введите имя для пользователя.", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);
            return;
        }

                    TrustedFacesList.Rows.Add(TrustedFacesList.Rows.Count + 1, FaceNameTextBox.Text.Trim(), "Удалить");

        //Converting image to base64 string and adding it to the list.
        ImageDataList.Add(ImageToBase64((Image)PreviewBox.Image.Clone()));
        //Adding name of the face to the list
        NameList.Add(FaceNameTextBox.Text.Trim());

        //Saves the face image data as a base encoded string, along with its name
        Properties.Settings.Default.Base64ImageData = ImageDataList;
        Properties.Settings.Default.FaceNames = NameList;
        Properties.Settings.Default.Save();

                    MessageBox.Show("Доверенное лицо включено", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);

        PreviewBox.Image = null;
        FaceNameTextBox.Text = "";
```

```
            FaceNameTextBox.Enabled = false;
            AddFaceButton.Enabled = false;
```

В қосымшасының жалғасы

```
NameLabel.Enabled = false;
        }

         private static Image CropImage(Image img, Rectangle cropArea) // Method that crops an image
        {
            Bitmap bmpImage = new Bitmap(img);
            return bmpImage.Clone(cropArea, bmpImage.PixelFormat);
        }

          private string ImageToBase64(Image image) // Method to convert an image to it's corresponding base ecoded string
        {
            using (var ms = new MemoryStream())
            {
                image.Save(ms, ImageFormat.Jpeg);
                return Convert.ToBase64String(ms.ToArray());
            }
        }

          private async void RingBell_Click(object sender, EventArgs e) // Method that verifies a face and performs door unlock, upon successfull verification.
        {
            if (!SecurityRunning) return;

            RingBell.Enabled = false;
            FrontDoorGB.Enabled = false;
            MonitorControls.Enabled = false;
            StatusLabel.Text = "Верификация..";
            StatusLabel.ForeColor = Color.Orange;

            //Plays ding dong caling bell sound
                        string FileName = string.Format("{0}Resources\\dingdong.mp3", Path.GetFullPath(Path.Combine(AppDomain.CurrentDomain.BaseDirectory, @"..\..\")));

                                    WMPLib.WindowsMediaPlayer wplayer = new WMPLib.WindowsMediaPlayer();
            wplayer.URL = FileName;
            wplayer.controls.play();
```

```
string currentImage = ImageToBase64((Image)Monitor.Image.Clone());
```

В қосымшасының жалғасы

```
 //Verbally greets or informs user with the verification result


            for (int i = 0; i < ImageDataList.Count; i++)// Iterates through each trusted face and compares it with current captured face
        {
            double confidence = await VerifyFace(ImageDataList[i], currentImage);

            if (confidence > 80) //Trusted face encountered, opening door.
            {
                //synthesizer.SpeakAsync("Welcome " + NameList[i] + ". Door is now unlocked.");
                synthesizer.SpeakAsync("Добро пожаловать " + NameList[i] + ". Дверь открыта.");
                StatusLabel.Text = "Открываю дверь...";
                StatusLabel.ForeColor = Color.Green;
                await Task.Delay(100);

                pictureBox2.Image = unlockedDoor;


                await Task.Delay(2500);

                StatusLabel.Text = "Приложение запущено!";
                RingBell.Enabled = true;
                FrontDoorGB.Enabled = true;
                MonitorControls.Enabled = true;
                return;
            }
        }

        //synthesizer.SpeakAsync("You're not authorized. Step away from the door.");
        synthesizer.SpeakAsync("Вы не авторизованы. Отойдите от двери!");

        StatusLabel.Text = "Вы не прошли верификацию!";
        StatusLabel.ForeColor = Color.Red;
        await Task.Delay(100);

        pictureBox2.Image = lockedDoor;

        await Task.Delay(2500);
```

```
            StatusLabel.Text = "Приложение запущено!";
            StatusLabel.ForeColor = Color.Green;
```

В қосымшасының жалғасы

```
            RingBell.Enabled = true;
            FrontDoorGB.Enabled = true;
            MonitorControls.Enabled = true;
        }

        private async Task UnlockDoor() // Method that performs door unlock operation
        {

        }

         private async void StartMonitorButton_Click(object sender, EventArgs e) // Method that intializes connection with the bolt device
        {
            StartMonitorButton.Enabled = false;
            FrontDoorGB.Enabled = false;

            StatusLabel.Text = "Подключение..";
            StatusLabel.ForeColor = Color.Orange;

            await Task.Delay(2500);

            SecurityRunning = true;
            StopMonitorButton.Enabled = true;
            StartMonitorButton.Enabled = false;
            FrontDoorGB.Enabled = true;

            StatusLabel.Text = "Приложение запущено!";
            StatusLabel.ForeColor = Color.Green;

            ListenForBell = true;
        }


          private void StopMonitorButton_Click(object sender, EventArgs e) //Method that stops camera motitoring. This wil terminate the bell ring listening thread.
        {
            SecurityRunning = false;
            FrontDoorGB.Enabled = false;
            StopMonitorButton.Enabled = false;
            StartMonitorButton.Enabled = true;
```

```
            StatusLabel.Text = "Приложение не запущено";
```

В қосымшасының жалғасы

```
 StatusLabel.ForeColor = Color.Red;

            ListenForBell = false;
        }

            private async void LockDoorButton_Click(object sender, EventArgs e) //
Method that gets called when user presses lock button
        {
            MonitorControls.Enabled = false;
            FrontDoorGB.Enabled = false;
            StatusLabel.Text = "Закрываю дверь..";
            synthesizer.SpeakAsync("Дверь закрыта!");
            StatusLabel.ForeColor = Color.Red;
            await Task.Delay(100);

            pictureBox2.Image = lockedDoor;

            await Task.Delay(2500);

            MonitorControls.Enabled = true;
            FrontDoorGB.Enabled = true;
            StatusLabel.Text = "Приложение запущено!";
            StatusLabel.ForeColor = Color.Green;
        }


          private void Form1_FormClosing(object sender, FormClosingEventArgs e) //
Methor that gets called when the application is exiting
        {

//Release the camera device before exiting
            CurrentCam.Stop();
        }
    }
}
```

Протокол анализа Отчета подобия Научным руководителем

Заявляю, что я ознакомился(-ась) с Полным отчетом подобия, который был сгенерирован Системой выявления и предотвращения плагиата в отношении работы:

Автор: Кәкіш Ерасыл

Название: Жеке тұлғаны тануға арналған веб-қосымшаны әзірлеу

Координатор: Дүйсенбаева Асемгуль

Коэффициент подобия 1: 2,80

Коэффициент подобия 2: 2,17

Замена букв: 0

Интервалы: 0

Микропробелы: 1

Белые знаки: 0

После анализа Отчета подобия констатирую следующее:

☐ обнаруженные в работе заимствования являются добросовестными и не обладают признаками плагиата. В связи с чем, признаю работу самостоятельной и допускаю ее к защите;

☐ обнаруженные в работе заимствования не обладают признаками плагиата, но их чрезмерное количество вызывает сомнения в отношении ценности работы, по существу, и отсутствием самостоятельности ее автора. В связи с чем, работа должна быть вновь отредактирована с целью ограничения заимствований;

☐ обнаруженные в работе заимствования являются недобросовестными и обладают признаками плагиата, или в ней содержатся преднамеренные искажения текста, указывающие на попытки сокрытия недобросовестных заимствований. В связи с чем, не допускаю работу к защите.

Обоснование:

Заимствования являются добросовестными и не обладают признаками плагиата.

24.05.2021

Дата *Подпись Научного руководителя*

Протокол анализа Отчета подобия
заведующего кафедрой / начальника структурного подразделения

Заведующий кафедрой / начальник структурного подразделения заявляет, что ознакомился(-ась) с полным отчетом подобия, который был сгенерирован Системой выявления и предотвращения плагиата в отношении работы:

Автор: Кәкіш Ерасыл

Название: Жеке тұлғаны тануға арналған веб-қосымшаны әзірлеу

Координатор: Дүйсенбаева Асемгуль

Коэффициент подобия 1: 2,80

Коэффициент подобия 2: 2,17

Замена букв: 0

Интервалы: 0

Микропробелы: 1

Белые знаки: 0

После анализа отчета подобия заведующий кафедрой / начальник структурного подразделения констатирует следующее:

- ☐ обнаруженные в работе заимствования являются добросовестными и не обладают признаками плагиата. В связи с чем, признаю работу самостоятельной и допускаю ее к защите;
- ☐ обнаруженные в работе заимствования не обладают признаками плагиата, но их чрезмерное количество вызывает сомнения в отношении ценности работы по существу и отсутствием самостоятельности ее автора. В связи с чем, работа должна быть вновь отредактирована с целью ограничения заимствований;
- ☐ обнаруженные в работе заимствования являются недобросовестными и обладают признаками плагиата, или в ней содержатся преднамеренные искажения текста, указывающие на попытки сокрытия недобросовестных заимствований. В связи с чем, не допускаю работу к защите.

Обоснование:

Заимствования являются добросовестными и не обладают признаками плагиата.

Дата 31.05.2021  Сейлова Н.А.

зав.кафедрой КБОиХИ