СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Преддипломная практика на предприятии даёт возможность молодому специалисту ознакомиться со специальным оборудованием и программным обеспечением, применяемом при разработке, сопровождении и эксплуатации информационных систем. Также, нахождение на предприятии специалиста под контролем руководителя позволит обрести и закрепить знания, и приобрести практический опыт работы.

Предприятие ОАО «Гомельский литейный завод «Центролит», является крупнейшим предприятием Беларуси, специализирующимся на производстве литых изделий из серого и высокопрочного чугуна.

Автоматизация – одно из направлений научно-технического прогресса, использующее саморегулирующие технические средства и математические методы с целью освобождения человека от участия в процессах получения, преобразования, передачи и использования энергии, материалов, изделий или информации, либо существенного уменьшения степени этого участия или трудоёмкости выполняемых операций.

Целью преддипломной практики является разработка мобильного приложения «Центролит Фото» для фотографии сотрудников предприятия ОАО ГЛЗ «Центролит» и получение опыта работы на предприятии.

Основные задачи, которые должны быть выполнены за период прохождения преддипломной практики:

– ознакомление с базой прохождения практики;

– изучение процесса хранения и передачи данных в организации;

– ознакомление со структурой организации, перечнем решаемых задач и внутренним распорядком;

– ознакомление с технологическим процессом обработки информации в подразделениях;

– изучение программного обеспечения, применяемого в организации;

– сбор информации согласно теме индивидуального задания;

–создание мобильного приложения согласно, индивидуального задания;

– закрепление знаний и навыков, полученных во время обучения.

1 ОБЩИЕ СВЕДЕНИЕ ОБ ОРГАНИЗАЦИИ ОАО «ГОМЕЛЬСКИЙ ЛИТЕЙНЫЙ ЗАВОД «ЦЕНТРОЛИТ»

1.1 История создания и развития организации

Открытое акционерное общество «Гомельский литейный завод «Центролит» – специализированный завод по изготовлению чугунного литья для станкостроения и машиностроения Постановлением ЦК КПБ и Совета Министров БССР решено было построить в Гомеле по проекту, разработанному Киевским институтом «Гипрохиммаш». Строительство его началось в 1963 г., а в 1965г. на площадях введенного в эксплуатацию ремонтно-механического цеха была получена первая продукция – нестандартное оборудование для дальнейшего развития собственного производства.

Производственная история Гомельского литейного завода «Центролит» началась в октябре 1968 года.

Первой продукцией завода было литье для многих типов станков, выпускаемых на станкозаводах РСФСР, УССР, БССР, ЛитССР, ЛатССР. Кооперированными поставками завод был связан со всеми крупными станко- и машиностроительными заводами СССР.

В 80-е годы прошлого столетия «Центролит» приступил к освоению производства чугунных тюбингов для «Минскметростроя».

Максимальный объем выпуска отливок был достигнут в 1988 году и составил 85,5 тысяч тонн.

Затем по причине экономического кризиса и падения спроса на литье выпуск упал до 7 тысяч тонн в 1995 году.

В 90-е годы, отмеченные годами застоя, руководящая команда завода сумела найти новые направления, способные вывести завод из тяжелой экономической ситуации. В это время приступили к выпуску дорожной арматуры и изделий для городского дизайна.

Сегодня предприятие изготавливает около 5 тысяч наименований продукции и продолжает наращивать объемы производства.

За годы работы ГЛЗ «Центролит» освоил сложное литье для машиностроения, станкостроения, автомобилестроения. Производственные возможности позволяют изготавливать литье массой от 1 кг до 19 тонн. За последнее десятилетие поставки продукции производились 400 предприятиям Беларуси, 70 предприятиям России, а также компаниям Финляндии, Франции, Италии, Венгрии, Германии, Турции, Армении, Азербайджана, Казахстана, Украины, Литвы, Латвии, Болгарии.

Республиканское унитарное предприятие «Гомельский литейный завод «Центролит» с 23.12.2010 года преобразовано в Открытое акционерное общество «Гомельский литейный завод «ЦЕНТРОЛИТ» (ОАО "ГЛЗ «ЦЕНТРОЛИТ»).

Основное производство предприятия расположено в 3 раздельных корпусах: литейный и плавильный корпус, корпус обрубки и очистки литья, а также черновой механической обработки, корпус хранения и подготовки шихтовых и формовочных материалов. Корпуса связаны между собой различными коммуникациями, сохранив тем самым производственную взаимосвязь.

Предприятие расположено на территории на юго-западной промышленной зоне г. Гомеля. В состав завода входят следующие блоки производственных зданий и цехов:

– корпус шихтовых и формовочных материалов;

– литейный корпус: цех плавки, цех крупного и среднего чугунного литья, цех мелкого чугунного литья;

– корпус обрубки и очистки отливок;

– блок вспомогательных цехов;

– ремонтно-механический, деревомодельный цех, инструментальный и литейный цех;

– энергоцех и транспортный цех;

– скрапобаза и склады ОМТС.

Органами управления предприятия является:

– общее собрание акционеров;

– наблюдательный совет;

– исполнительные органы (дирекция и директор).

Парк технологического оборудования ОАО ГЛЗ «Центролит» включает в себя 273 единицу металлорежущего оборудования, 36 единиц кузнечнопрессового, 436 литейного и ряд другого оборудования. Доля оборудования с возрастом свыше 20 лет составляет 44% общего числа единиц оборудования.

Доля оборудования с возрастом от 15 лет до 20 лет составляет 20,6% общего числа единиц оборудования; от 10 лет до 15 лет – 23,2% общего числа единиц и лишь 8% оборудования с возрастом до 5 лет.

В соответствии с законодательством ОАО ГЛЗ «ЦЕНТРОЛИТ» осуществляет следующие виды деятельности:

– литьё металлов;

– производство строительных металлических конструкций;

– обработка металлов и нанесение покрытий на металлы;

– обработка металлических изделий с использованием основных технологических процессов машиностроения;

– неспециализированная оптовая торговля непродовольственными товарами;

– розничная торговля в неспециализированных магазинах;

– предоставление услуг гостиницами без ресторанов;

– производство спецодежды;

– распиловка и строгание древесины;

– производство столярных изделий;

– издание газет;

– производство сборных железобетонных и бетонных конструкций и изделий;

– производство прочей мебели;

– переработка отходов и лома черных металлов;

– переработка отходов и лома цветных металлов;

– предоставление услуг столовыми при предприятиях и учреждениях;

– деятельность в области архитектуры, инженерные услуги;

1.2 Используемые технологии в отделе автоматизированных систем управления предприятием (ОАСУП)

За время прохождения преддипломной практики на базе организации ОАО «Гомельский литейный завод «Центролит» в отделе автоматизированных систем управления предприятием был получен опыт работы с различными технологиями и программным обеспечением, таким как:

PostgreSQL – это объектно-реляционная система управления базами данных (ОРСУБД), основанная на *POSTGRES, Version* 4.2 – программе, разработанной на факультете компьютерных наук Калифорнийского университета в Беркли. В *POSTGRES* появилось множество новшеств, которые были реализованы в некоторых коммерческих СУБД гораздо позднее.

PostgreSQL – СУБД с открытым исходным кодом, основой которого был код, написанный в Беркли. Она поддерживает большую часть стандарта SQL и предлагает множество современных функций:

– сложные запросы;

– внешние ключи;

– триггеры;

– изменяемые представления;
– транзакционная целостность;
– многоверсионность;

Кроме того, пользователи могут всячески расширять возможности *PostgreSQL*, например создавая свои:

– типы данных;
– функции;
– операторы;
– агрегатные функции;
– методы индексирования;
– процедурные языки;

А благодаря свободной лицензии, *PostgreSQL* разрешается бесплатно использовать, изменять и распространять всем и для любых целей – личных, коммерческих или учебных.

Lazarus – свободная среда разработки программного обеспечения с открытым исходным кодом, которая построена на компиляторе *Free Pascal* с добавлением Интегрированной Среды Разработки. *Lazarus* является многоцелевым инструментом программирования, то есть на нем можно создавать программы различных типов (консольные приложения, динамически-подгружаемые библиотеки, GUI приложения).

Lazarus содержит в себе редактор кода, визуальный проектировщик форм, а также библиотеку компонентов, которая очень хорошо совместима с Библиотекой Визуальных Компонентов *Delph*i (*VCL*). Библиотека Визуальных Компонентов *Lazarus* (*LCL*) включает эквиваленты для большинства контролов из *VCL* (формы, кнопки, текстовые поля и т.д.), которые используются для создания приложений с графическим интерфейсом.

Embracadero Delphi, ранее *Borland Delphi* и *CodeGear Delphi* – интегрированная среда разработки ПО для *Microsoft Windows*, *Mac OS* , *iOS* и *Android* на языке Delphi (ранее носившем название *Object Pascal*), созданная превоначально фирмой Borland и на данный момент принадлежащая и разрабатываемая *Embarcadero Technologies*. *Embarcadero Delphi* является частью пакета *Embarcadero RAD Studio* и поставляется в пяти редакциях: *Community* (распространяется бесплатно и имеет ограниченную лицензию на использование в коммерческих целях), *Professional*, *Enterprise* и *Architect*. Координирующий офис *Embarcadero*, ответственный за разработку *Delphi*, находится в Торонто, тогда как сама разработка сконцентрирована главным образом в Канаде и Испании.

PHP – скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.

Язык и его интерпретатор *Zend Engine* разрабатываются группой энтузиастов в рамках проекта с открытым кодом. Проект распространяется под собственной лицензией, несовместимой с *GNU GPL*.

Apache – это программное обеспечение с открытым исходным кодом, веб-сервер, который обеспечивает работу около 46% сайтов по всему миру. Официальное название – *Apache HTTP* Сервер, поддерживается и развивается компанией *Apache Software Foundation*.

Веб-сервер позволяет владельцам сайтов обслуживать их контент в интернете, о чём понятно с самого название «веб-сервер». *Apache* один из самый старых и надёжный веб-серверов с первой версией выпуска более 20 лет назад в 1995 году.

1.3 Охрана труда и техника безопасности на предприятии

В начале прохождения преддипломной практики на предприятии необходимо пройти инструктаж по технике безопасности и охране труда.

На всех предприятиях и организациях обеспечение здоровья и безопасности условий труда возлагается на администрацию. Обязанности работодателя по созданию здоровых и безопасных условий труда закреплены в положениях о предприятиях, в коллективных договорах, в правилах внутреннего трудового распорядка.

Учащийся обязан изучить и строго соблюдать в дальнейшем положения инструктажа.

Целью инструктажа является ознакомление с необходимыми знаниями по охране труда.

По характеру и времени проведения инструктаж по охране труда подразделяют на:

– вводный;

– первичный на рабочем месте;

– повторный;

– внеплановый;

– целевой.

Вводный инструктаж проводится с целью ознакомления работника, учащегося или студента, прибывшего на производственное обучение со спецификой работы организации и общими требованиями по охране труда.

На предприятии инструктаж проводит инженер по охране труда или лицо, на которое возложены эти обязанности. На крупных предприятиях к проведению разных частей инструктажа могут быть привлечены соответствующие специалисты. В журнале регистрации вводного инструктажа по охране труда и на контрольном листе делают запись о проведении инструктажа с обязательной подписью того, кто получил инструктаж.

В день начала преддипломной практики, мастером по охране труда был проведен вводный инструктаж по охране труда и технике безопасности на ОАО ГЛЗ «Центролит».

Также, требовалось обязательно ознакомиться с правилами внутреннего трудового распорядка и при необходимости оформить пропуск.

Организация работает с понедельника по пятницу и посменно.

Учащиеся или студенты, находящиеся на практике, посещают предприятие в соответствии со следующими правилами внутреннего распорядка:

– время работы: понедельник – пятница с 8:00 до 16:30;

– обеденный перерыв: с 11:40 до 12:10.

Требования охраны труда при прохождении производственной технологической практики:

Охрана труда – это система или комплекс мероприятий, направленных на защиту здоровья работника в процессе его трудовой деятельности.

Основные мероприятия охраны труда делятся на группы:

- правовые;
- социально-экономические;
- организационно-технические;
- санитарно-гигиенические.
- профилактические.

Порядок проведения инструктажей по охране труда установлен в Инструкции о порядке обучения, стажировки, инструктажа и проверки знаний, работающих по вопросам охраны труда, утвержденной постановлением мин. труда и соц. защиты.

Помещения, предназначенные для размещения рабочих мест, оснащенных персональными компьютерами, следует оснащать солнцезащитными устройствами. Все помещения с персональными

компьютерами должны иметь естественное и искусственное освещение. Для борьбы с запыленностью воздуха необходимо проводить влажную ежедневную уборку и регулярное проветривание помещения.

Требования безопасности перед началом работы:

- осмотреть и привести в порядок рабочее место;
- отрегулировать освещенность на рабочем месте, убедиться в достаточности освещенности, отсутствии отражений на экране, отсутствии встречного светового потока;
- проверить правильность подключения оборудования в электросеть;
- протереть специальной салфеткой поверхность экрана;
- убедиться в отсутствии дискет в дисководах процессора персонального компьютера;
- Требования безопасности во время работы:
- соблюдать требования охраны труда, установленные настоящей Инструкцией;
- расстояние от глаз оператора до экрана должно быть в пределах 60-80 см;
- местный источник света по отношению к рабочему месту должен располагаться таким образом, чтобы исключить попадание в глаза прямого света, и должен обеспечивать равномерную освещенность на поверхности 40х40 см, не создавать слепящих бликов на клавиатуре и других частях пульта, а также на экране видеотерминала в направлении глаз работника;
- для снижения зрительного и общего утомления после каждого часа работы необходимо делать перерывы;
- содержать в порядке и чистоте свое рабочее место;
- держать открытыми вентиляционные отверстия оборудования;

В области охраны труда инженер-программист обязан:

участвовать в реализации политики и целей в области охраны труда в пределах своих полномочий;

– соблюдать требования по охране труда и пожарной безопасности, а также правила поведения на территории организации, в аудиториях, вспомогательных и бытовых помещениях;

– использовать и правильно применять средства индивидуальной защиты;

– проходить в установленном законодательством порядке медицинские осмотры, обучение, инструктаж и проверку знаний по вопросам охраны труда;

– заботиться о личной безопасности и личном здоровье, а также о безопасности окружающих в процессе выполнения работ либо во время нахождения на территории университета;

– немедленно сообщать непосредственному руководителю о любой ситуации угрожающей жизни или здоровью работающих и окружающих, несчастном случае, произошедшем на производстве, оказывать содействие непосредственному руководителю в принятии мер по оказанию необходимой помощи потерпевшим и доставке их в организацию здравоохранения;

– выполнять нормы и обязательства по охране труда, предусмотренные коллективным договором, соглашением, трудовым договором, правилами внутреннего трудового распорядка, должностными обязанностями;

– в случае отсутствия средств индивидуальной защиты немедленно уведомлять об этом непосредственного руководителя либо иного уполномоченного должностного лица нанимателя;

– оказывать содействие и сотрудничать с непосредственным руководителем в деле обеспечения здоровых и безопасных условий труда, немедленно извещать своего непосредственного руководителя или иного уполномоченного должностного лица нанимателя о неисправности оборудования, приспособлений, средств защиты, об ухудшении состояния своего здоровья;

– поддерживать свое рабочее место, оборудование и приспособления в исправном состоянии, порядке и чистоте;

– исполнять другие обязанности, предусмотренные законодательством об охране труда.

2 СТРУКТУРА РАЗРАБОТАННОГО МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ФОТОГРАФИИ СОТРУДНИКОВ ОАО «ГОМЕЛЬСКИЙ ЛИТЕЙНЫЙ ЗАВОД «ЦЕНТРОЛИТ»

2.1 Предметная область разработанного мобильного приложения

За время прохождения преддипломной практики на базе ОАО «Гомельский литейный завод «Центролит» в отделе автоматизированных систем управления предприятием (ОАСУП) было разработано мобильного приложение «CentrolitPhoto».

Данное приложение выполняет несколько функций согласно заданию, полученному от руководителя практики от предприятия:

- Выполнять фотографию сотрудника в формате 4x3;
- Сохранять три копии фотографии в следующих разрешениях (640x480, 800x600, 1920x1080);
- Используя FTP соединение выполнять копирование всех фотографий из каталога на FTP-сервер отдела АСУП;
- Используя FTP соединение выполнять копирование фотографий по табельному номеру сотрудника на FTP-сервер отдела АСУП;
- Очистка каталога по нажатию клавиши «Очистить каталог»

Данные фотографии будут хранится в единой базе данных сотрудников и использоваться в документациях и приложениях, где необходима фотография сотрудника.

По результату практики данное приложение было закончено и использовано в системе автоматизации расчёта сотрудников за обеды в столовой. А именно, сотрудник имеет право записывать стоимость обеда в счёт будущей заработной платы, называя свой табельный номер. При этом кассир вводит табельный номер сотрудника и видит размер его кредита и фотографию, чтобы предотвратить превышения суммы или же мошенничества (записать обед на чужое имя).

Тем самым можно сказать о том, что приложение получило реальное внедрение в производство.

Аналогом разработанного приложения можно считать приложение «Фото на паспорт».

С помощью этого приложения возможно мгновенно отформатировать фотографий для паспорта и распечатать или сохранить их.

Интегрированные шаблоны для паспортных фотографий из 100 стран мира позволяет Вам регулировать точную высоту головы и центральное расположения по горизонтали.

Современные мобильные устройства, такие как *iPhone* и *iPad* являются идеальными для создания паспортных фотографий с их встроенной камерой.

Приложение под названием "Фото на паспорт", позволит Вам сохранить несколько паспортных фотографии на одном формате в JPEG, напечатать или отправить по электронной почте.

Фотографии могут быть распечатаны непосредственно или сохранены для будущего заказа в любой фотолаборатории в интернете.

Для получения паспортных фотографий изображение фиксируется на камеру или производится загрузка сохраненного изображения. При необходимости Вы можете исправить, насыщенность, яркость и контрастность.

Поддерживаемые форматы: 3.5x4.5, 9x13, 10x13, 10x15, 11x15, 13x18

Это приложение поддерживает печать на совместимом принтере *AirPrint*. Паспортная фотография печатается автоматически немного больше для более удобного вырезания. В случае необходимости, размер печати можно отрегулировать в настройках приложения.

Однако это приложение является широко направленным, в отличие от разработанного за время практики приложения.

2.2 Особенности разработки мобильных приложений под операционную систему Android

Операционная система *Android* была выпущена компанией *Google* 23 сентября 2008 года. 11 июля 2005 года корпорация купила стартап *Android Inc*, занимающейся этой разработкой, и превратила это направление в одно из ключевых. С тех пор *Android* быстро развивается и теперь установлен на более чем 83% всех мобильных устройств в мире.

Одна из особенностей – это большая фрагментация устройств. Просто огромная. Это прекрасно для пользователей: можно выбрать телефон на любой вкус и под любые технические требования. Но очень непросто для разработчиков, и это касается как аппаратной, так и программной части.

Аппаратно устройство может иметь фронтальную камеру, может и нет. Симкарт может быть любое количество. Физические кнопки могут присутствовать или нет. Экрана может быть два: дополнительный с тыльной стороны или на чехле.

Существующие элементы также имеют разные параметры. Например, датчик акселерометра может быть установлен по-разному у разных устройств как показано на рисунке 2.1. [2]

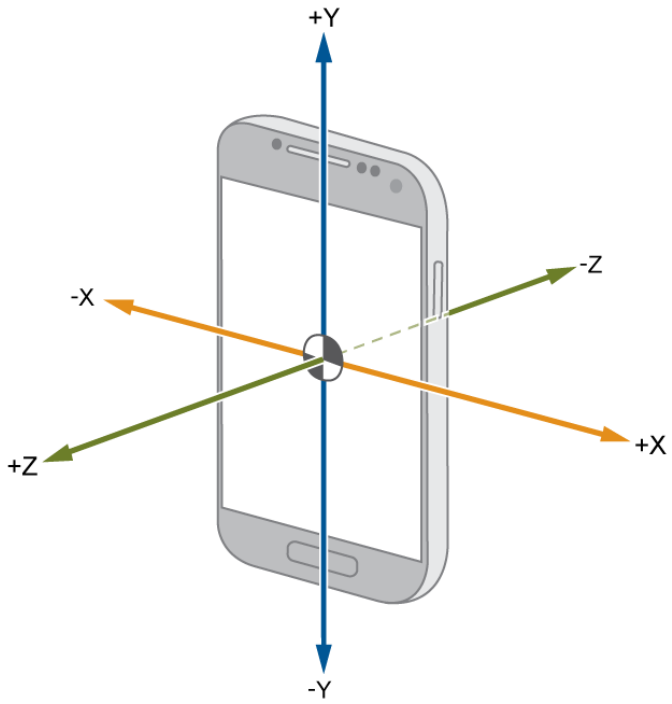

Рисунок 2.1 – Расположение осей акселерометра может оказаться не таким, как в документации

Размер экрана и его разрешение – отдельная проблема. Например, если вам нужно разместить изображение на весь экран *iOS* устройства, вы используете несколько изображений под типовые размеры *iPhone* 6 и выше, *iPhone* 6 *Plus* и выше, *iPhone X* и *iPhone X Max*. В случае же с *Android* экраны имеют и разное разрешение, и разное соотношение сторон, и разную плотность. Это отображено на рисунке 2.2. [1]

В связи с этим для *Android*–разработчиков существуют разные инструменты, например 9 *Patch* – схема разметки, позволяющая задать правила растягивания изображения при изменении его размера. Без нее сложно корректно отобразить в том числе и фоновые изображения в связи с разными размерами экранов.

Ещё одной проблемой является архитектура самого приложения. В отличие от *iOS*, где приложения архитектурно представляют собой нечто единое целое, в *Android* они собираются из логически самостоятельных и обособленных частей – активити и фрагментов. В отличие от *iOS*, мобильные

приложения для *Android* представляют собой взаимосвязь отдельных, логически обособленных элементов, как об этом говорилось выше.

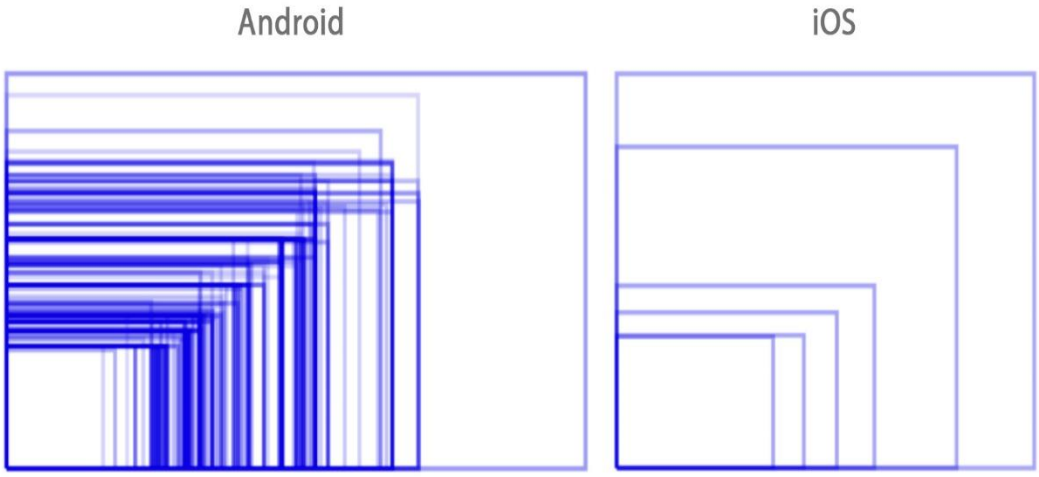

Рисунок 2.2 – Схематичная демонстрация основных размеров экранов *Android* и *iOS*

Соответственно если вы задались целью заполнить изображением весь экран, вам нужно или использовать несколько картинок и обрезать их на нестандартных размерах экранов, или разрезать так, чтобы отдельные части образовывали собой одно целое, но могли смещаться относительно друг друга (например, земля, облака, левая и правая части).[3]

Ещё одной проблемой является большой разброс версий операционной системы *Android*, установленных у пользователей. Это порождает множество проблем:

- Необходимо учитывать особенности отображение интерфейса на разных версиях ОС и разных оболочках. Так, системные элементы управления могут выглядеть совершенно по-разному на разных версиях *Android* и разных оболочках одной и той же версии *Android*;
- Разные версии в ряде моментов имеют разную логику работы. Например, до версии 6.0 приложения не должны были запрашивать каждое разрешение отдельно (доступ к камере, микрофону и так далее), они указывались списком в *Google Play* и, подразумевалось, что пользователь ознакамливается с ними до момента загрузки. Начиная с 6.0 каждое разрешение должно быть запрошено отдельно уже в момент работы приложения. Соотвественно, если вы не реализовываете оба варианта логики, приложение не будет работать либо до версии 6.0, либо в более поздних;[4]

– Программные методы и библиотеки меняются: какие-то из них признаются устаревшими и их требуется заменять на более новые. Таким образом всегда встает выбор: либо поддерживать наиболее последние функции ОС, либо позволить, как можно большему количеству пользователей установить мобильное приложение;
– В последних версиях ОС добавилась многозадачность рабочей области. Пользователь может отобразить на экране одновременно несколько приложений и вашему может быть выделена совершенна произвольная по размеру область. Это также надо учитывать.

Ещё одной проблемой является архитектура самого приложения. В отличие от *iOS*, где приложения архитектурно представляют собой нечто единое целое, в *Android* они собираются из логически самостоятельных и обособленных частей – активити и фрагментов.[5]

Такой подход был выбран как раз для того, чтобы обеспечить работу приложений на совершенно разных устройствах, в том числе с очень малым объемом оперативной памяти и очень слабыми процессорами. Если части приложения независимы, любую из них можно в нужный момент выбросить из памяти и не тратить на поддержание ее жизненного цикла драгоценные ресурсы.

Например, вы видите на экране список ресторанов, затем нажимаете на какой-то элемент и проваливаетесь в него. Второй экран, карточка ресторана, ничего не должна знать о предыдущем экране, списке, потому что в любой момент времени, в том числе сразу после перехода в карточку, он может быть выгружен из оперативной памяти и уничтожен. Это произойдет, например, если в фоне запущенно много приложений или на экране карточки вы начинаете проигрывать видео в хорошем качестве.

Чтобы приложение работало корректно и без сбоев, экран карточки не должен обращаться ни к какой информации предыдущего экрана, принимая на вход лишь определенные данные. Если, например, у пользователя есть возможность перейти на следующий ресторан, не возвращаясь в список, то карточка должна самостоятельно получить необходимую информацию. В то же время экран списка не должен ничего знать о самой карточке, т.к. после возвращения из нее она так же может быть уничтожена.

Этот аспект архитектуры приложений звучит слишком технически, но он дает понять почему, например, далеко не все типы приложений возможно реализовать кроссплатформенно: если это что-то объемное по функционалу, то оно полностью выгружается из памяти при недостатке места и на слабых устройствах возможность работы с ними попросту отсутствует.

В отличие от *iOS*, мобильные приложения для *Android* представляют собой взаимосвязь отдельных, логически обособленных элементов, как об этом говорилось выше. То есть нельзя просто взять и портировать приложение на другую мобильную операционную систему, переписав код с одного языка программирования на другой. Нужно закладывать совершенно другую архитектуру.

Другой подход наблюдается и в других аспектах. Например, современная иконка приложения может иметь разную форму в зависимости от настроек операционной системы. Дизайнер должен это учитывать и убедиться, что логотип выглядит прекрасно и гармонично во всех вариантах.

Тестирование на большом парке физических устройств (не эмуляторов) при этом имеет чрезвычайно важное значение. Даже оно в силу огромного количества разных телефонов на рынке не обеспечивает беспроблемное функционирование на всех доступных моделях, но по крайней мере снижает вероятность проблем на наиболее популярных девайсах.

Перед публикацией в магазин приложений *Google Play* сборки проходят гораздо менее тщательный контроль с точки зрения соблюдения требований по построению интерфейса, выбора тематики и запроса персональных данных пользователей.

Несмотря на то, что *Google* недавно изменил подход к проверке приложений, сделав его более тщательным и более ручным, среднее время ревью приложений составляет 2-4 часа. Это существенно быстрее, чем 2-3 дня в случае с *App Store*. Схема процесса проверки приложений представлена на рисунке 2.3.

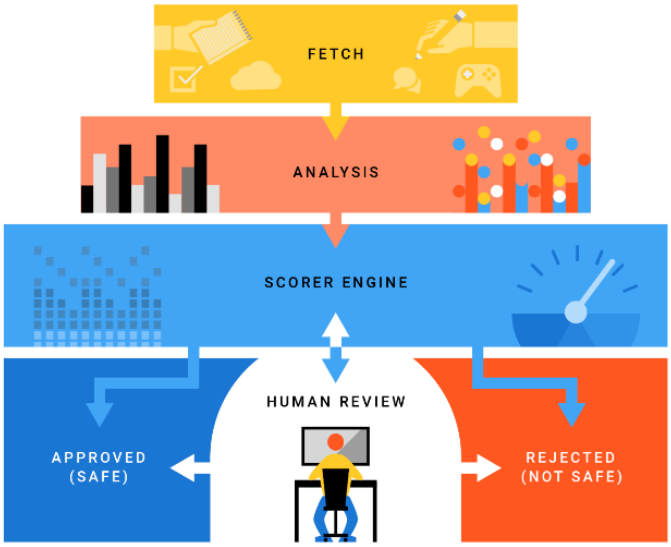

Рисунок 2.3 – Схема процесса проверки приложений

Раньше многие этим пользовались с целью раскрутки приложения. В *Android Market*, предыдущей версии *Google Play*, была вкладка «Новые приложения», отображающая обновления в их хронологической последовательности. Таким образом, чем чаще вы обновляли версию своего приложения, тем больше получали загрузок. Каждый апдейт в России давал 1-3 тысячи загрузок, и за месяц можно было достичь хороших результатов, если приложение действительно приносило пользу (если рейтинги плохие, загрузок вы получали единицы).[6]

Сейчас продвижение приложений стало намного сложнее, но в первую очередь нужно сделать классный продукт, в том числе с точки зрения требований концепции *Material Design*. Вы их можете не соблюдать, но такое приложение будет рекламироваться в *Google Play* самой *Google* в исключительных случаях.

2.3 Технологии и средства, использованные при разработке мобильного приложения

Во время выполнения задания по преддипломной практике было разработано мобильное приложение на базе *Android* для фотографирования сотрудников организации ОАО «Гомельский литейный завод «Центролит».

Для этого были использованы следующие технологии.

Android SDK – универсальное средство разработки мобильных приложений для операционной системы *Android*. Отличительной чертой от обычных редакторов для написания кодов является наличие широких функциональных возможностей, позволяющих запускать тестирование и отладку исходных кодов, оценивать работу приложения в режиме совместимости с различными версиями ОС *Android* и наблюдать результат в реальном времени. Поддерживает большое количество мобильных устройств, среди которых выделяют: мобильные телефоны, планшетные компьютеры, умные очки (в том числе *Google Glass*), современные автомобили с бортовыми компьютерами на ОС Android, телевизоры с расширенным функционалом, особые виды наручных часов и многие другие мобильные гаджеты, габаритные технические приспособления.

Android API – Расшифровывается как *Application Programming Interface* (программный интерфейс приложения). Это просто интерфейс, уровень абстракции, который обеспечивает связь между двумя разными «частями» программного обеспечения. Он работает как договор между поставщиком (например, библиотекой) и потребителем (например, приложением).

Это набор формальных определений, таких как классы, методы, функции, модули, константы, которые могут использоваться другими разработчиками для написания своего кода. При этом API не включает в себя реализацию

API Level это просто целое число, которое однозначно идентифицирует ревизию рабочего окружения программных библиотек (*framework API revision),* которая предоставляется версией платформы (операционной системой) *Android.*

Платформа *Android* предоставляет *framework API*, которое могут использовать приложения для взаимодействия с нижележащей системой *Android. Framework API* состоит из:

Базового набора пакетов и классов.

- Набора элементов *XML* и атрибутов для декларирования в файле манифеста.
- Набора элементов *XML* и атрибутов для декларирования и доступа к ресурсам.
- Набора разрешений доступа к ресурсам устройства и системы, которые приложение может запросить, а также защита ограничения доступа, встроенная в систему.
- Набора намерений.

Каждая последующая версия платформы *Android* может включать в себя обновления предоставляемого *framework API* приложения *Android*.

Обновления *framework API* разработаны так, чтобы новый API оставался совместимым со всеми более старыми версиями *API*. Т. е. большинство изменений в *API* добавочные, и представляют новый функционал, или функционал, заменяющий старый.

Поскольку части *API* были обновлены, то старые заменяемые части объявляются устаревшими и нежелательными для использования, но эти старые части все же не удаляются, чтобы существующие приложения могли продолжать их использовать. В очень редких случаях части *API* могут быть модифицированы или удалены, но только в том случае, если это необходимо для обеспечения устойчивости *API* и безопасности приложения или системы. Все другие части *API* от старых ревизий переносятся в новые ревизии платформы без модификации.

Framework API, который поставляет платформа *Android*, указывается в виде целого числа, и этот идентификатор в виде целого числа называется "*API Level*". Каждая версия платформы *Android* поддерживает точно один *API Level*, несмотря на неявную поддержку всех более старых версий API Level

(вплоть до A*PI Level 1*). Первоначальный релиз платформы *Android* предоставлял *API Level* 1 и с каждым новым релизом номер *API Level* последовательно увеличивался на единицу.

Embracadero Delphi, ранее *Borland Delphi* и *CodeGear Delphi* – интегрированная среда разработки ПО для *Microsoft Windows*, *Mac OS* , *iOS* и *Android* на языке Delphi (ранее носившем название *Object Pascal*), созданная превоначально фирмой Borland и на данный момент принадлежащая и разрабатываемая *Embarcadero Technologies*. *Embarcadero Delphi* является частью пакета *Embarcadero RAD Studio* и поставляется в пяти редакциях: *Community* (распространяется бесплатно и имеет ограниченную лицензию на использование в коммерческих целях), *Professional*, *Enterprise* и *Architect*.

FireMonkey (*FMX*) – *GUI*-фреймворк, использующий возможности графического процессора. Является кроссплатформенным: поддерживаются *Windows, Mac OS, Apple iOS* и *Android*.

Под названием *VG-Scene* разрабатывалась Евгением Крюковым из Улан-Удэ, Россия компания *KSDev* как вектор-базированная GUI-библиотека следующего поколения. В 2011 году американская компания *Embarcadero Technologies* купила права на библиотеку и включила её в состав своих продуктов. Позже Евгений Крюков получил работу в *Embarcadero Technologies.*

FireMonkey входит, параллельно с традиционной *Visual Component Library*, в состав *Delphi XE3, Delphi XE2, C++Builder XE2 и RAD Studio XE2*, включая *RadPHP и Embarcadero Prism*.

В 2012 году FireMonkey под кодовым именем FireMonkey FM2 вошла в состав *Delphi XE3, C++Builder XE3* и *RAD Studio XE3, Embarcadero HTML5 Builder и Embarcadero Prism XE3*.

В апреле 2013 года вышел *FireMonkey FM3*, которая распространяется вместе с *Embarcadero RAD Studio XE4*.

Используя возможности *Pixel Shader* 2.0 *FireMonkey* позволяет обогащать графический интерфейс программ широким набором визуальных эффектов. Даёт возможность строить пользовательские масштабируемые векторные и 3D-интерфейсы.

Позволяет отделить пользовательский интерфейс от бизнес-логики и механизмов доступа к данным, что, в частности, позволяет размещать невизуальные части приложения в облачных сервисах: *Amazon* или *Azure*.

Если попытаться в двух словах описать основную часть продукта – то это сценарный компонент (к примеру – главный компонент от *GLScene*), векторный, только 2D. Имеются свои компоненты, а главное – мощное

средство их редактирования со скинами. То есть с помощью встроенного редактора вы можете создать свой уникальный компонент, на базе одного или нескольких базовых. Имеются встроенные разнообразные методы анимирования компонентов, работа с векторной графикой.

FTP (*File Transfer Protocol*) – протокол передачи файлов по сети, является одним из старейших прикладных протоколов, появившихся задолго до *HTTP*, и даже до *TCP/IP*, в 1971 году; в первое время он работал поверх протокола *NCP*. Он и сегодня широко используется для распространения ПО и доступа к удалённым хостам. В отличие от *TFTP*, гарантирует передачу (либо выдачу ошибки) за счёт применения квотируемого протокола.

Протокол построен на архитектуре «клиент-сервер» и использует разные сетевые соединения для передачи команд и данных между клиентом и сервером. Пользователи *FTP* могут пройти аутентификацию, передавая логин и пароль открытым текстом, или же, если это разрешено на сервере, они могут подключиться анонимно. Можно использовать протокол *SSH* для безопасной передачи, скрывающей (шифрующей) логин и пароль, а также шифрующей содержимое.

2.4 Структура разработанного мобильного приложения

В разработанном мобильном приложении существует две вкладки: «Фотография» и «Копирование». Вкладка «Фотография» предназначена непосредственно для фотографии сотрудника и привязки этой фотографии к табельному номеру. Вкладка «Фотография» изображена на рисунке 2.4.

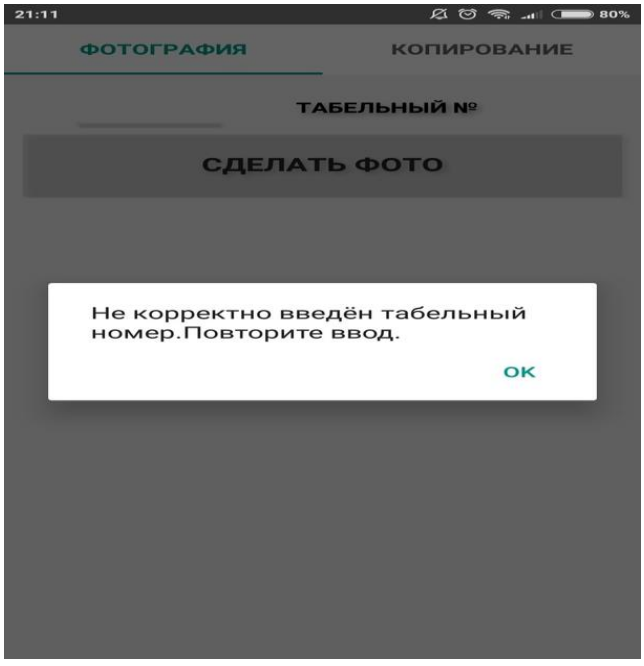

Рисунок 2.4 – Вкладка «Фотография»

Для того чтобы сделать фотографию необходимо ввести табельный номер, при его некорректном вводе будет выведено об этом сообщение.

После фотографирования сотрудника на вкладке «Фотография» будет отображён результат и выведено сообщение об успешном выполнении, как показано на рисунке 2.5.

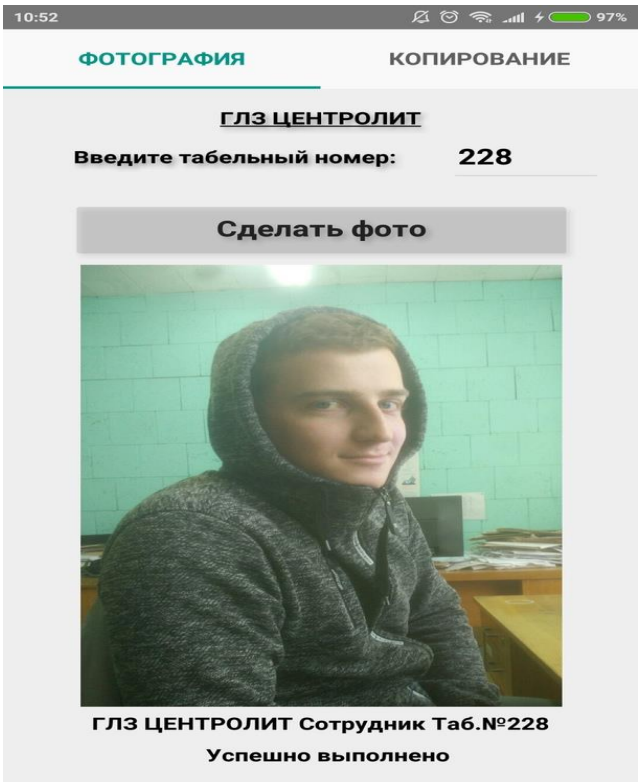

Рисунок 2.5 – Результат выполнения фотографии сотрудника

После этого фотография автоматически сохраняется в выбранный каталог в трёх копиях с разрешением 640x480, 800x600, 1920x1080 соответственно.

Вкладка «Копирование» предназначена для копирования уже сделанных фотографий из каталога на FTP-сервер.

При нажатии клавиши «Копировать всё» происходит копирование всех фотографий из каталога.

При нажатии клавиши «Копировать по таб.№» происходит копирование фотографий с определённым табельным номером, который определяется при вводе его в определённое поле.

При нажатии клавиши «Очистить каталог» происходит полная очистка каталога.

Если же каталог уже пуст, то об этом выводится сообщение как показано на рисунке 2.6.

Если же при копировании каталог пуст, то об этом так же выводится предупреждающее сообщение.

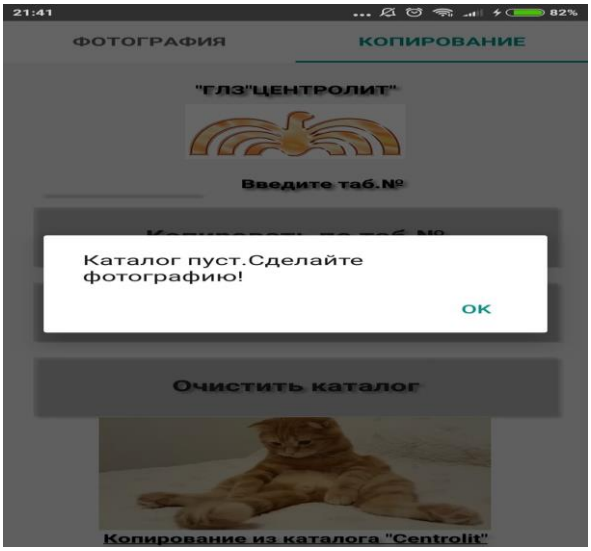

Рисунок 2.6 – Предупреждение о том, что каталог пуст

Если каталог пуст, то необходимо сделать фотографию и продолжить работу.

Следует заметить, что при копировании фотографий происходит их автоматическая замена, тем самым пользователь застрахован от случайного использования ненужных ему файлов.

При успешном копировании файлов соответственно выводится сообщение об этом, как показано на рисунке 2.7.

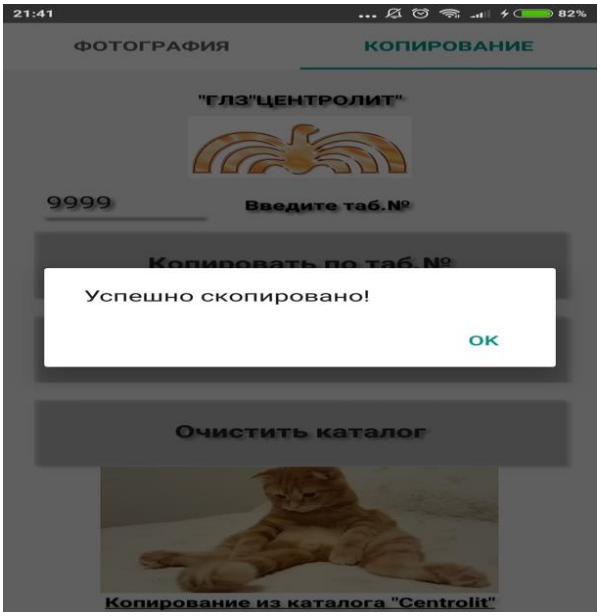

Рисунок 2.7 – Сообщение об успешном копировании файлов

При копировании по табельному номеру, в случае отсутствия фотографии с заданным номером, соответственно выводится предупреждающее сообщение об отсутствии файла с заданным номером.

Как показано на рисунке 2.8.

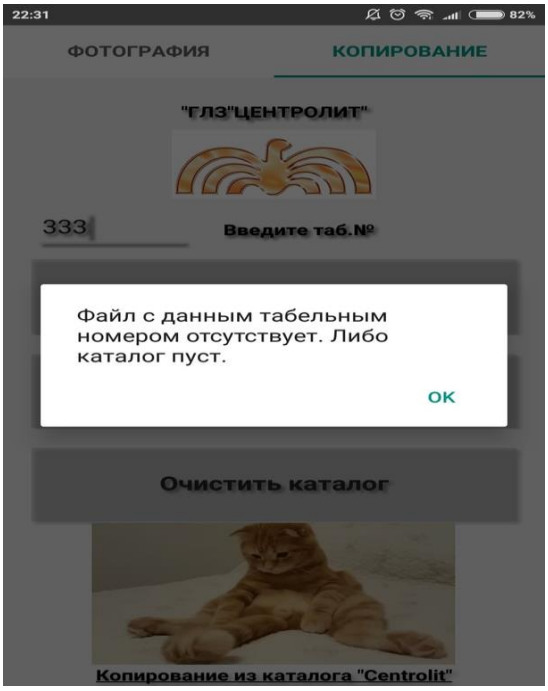

Рисунок 2.8 – Предупреждение об отсутствии файла с заданным табельным номером

Данное приложение застраховано от всех возможных некорректных способов ввода информации.

Также существует две версии приложения. Одна из которых работает только в пределах внутренней сети отдела автоматизации управления предприятием.

Вторая же версия приложения имеет открытый доступ к FTP-серверу отдела автоматизации управления предприятием из любой сети интернет.

ЗАКЛЮЧЕНИЕ

В процессе прохождения преддипломной практики были изучены задачи и обязанности инженера-программиста на предприятии ОАО ГЛЗ «Центролит».

Преддипломная практика предоставила возможность закрепить знания, которые были получены в ходе обучения, а также получить дополнительные специализированные знания.

В ходе прохождения учебной технологической практики были тщательно разобраны некоторые вопросы, такие как:

- Функции и задачи инженера-программиста;
- способы хранения данных;
- должностная инструкция;
- перечень производимых товаров;
- ПО, используемое на предприятии;
- структура предприятия;
- распорядок дня;
- инструкция по охране труда;
- материально-техническая база.

По истечению срока практики было разработано мобильное приложение, которое позволяет фотографировать сотрудников, а также переносить эти фотографии в единую базу данных сотрудников ОАО ГЛЗ «Центролит».

Данное приложение позволяет облегчить работу с сотрудниками в любой сфере учёта сотрудников, так как позволяет визуализировать информацию по каждому отдельному сотруднику

Интерфейс данного приложения очень прост в использовании, весь функционал становится понятен буквально за несколько минут.

Приложение разработано таким образом, что его будет поддерживать подавляющее большинство современных мобильных устройств, использующих операционную систему *Android*.

Данное приложение было разработано и внедрено непосредственно для предприятия ОАО ГЛЗ «Центролит».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Осипов, Д.Л. Delphi Программирование для Windows, OS X, iOS и Android / Д.Л. Осипов. – СПб.: БХВ-Петербург, 2014. — 464 с.

2. Эйдлин, Г.М. Delphi: программирование в примерах и задачах. / Г.М. Эйдлин. – СПб: Питер, 2018. – 138 с.

3. Дарвин, Я.Ф. Android. Сборник рецептов. Задачи и решения для разработчиков приложений. / Я.Ф. Дарвин. – Вильямс, 2017. – 768 с.

4. Клифтон, Я. Проектирование пользовательского интерфейса в Android. / Я. Клифтон – ДМК Пресс, 2017. – 452 с.

5. Блог компании Embracadero. Разработка кроссплатформенных мобильных приложений в Delphi [Электронный ресурс]. Режим доступа: *https://habr.com.* – Дата доступа: 19.03.2020

6. Рудольф, А.С. Android.Программирование для мобильных устройств. / А.С. Рудольф. – Вильямс, 2016. – 554 с.

7. Васильев, Д.Г. Программирование Delphi / Д.Г. Васильев. – СПб.: БХВ-Петербург, 2016. — 320 с.

ПРИЛОЖЕНИЕ А

(Обязательное)

Исходный код программы «Centrolit Photo»

```
unit Unit1;

interface

uses
  System.SysUtils, System.Types, System.UITypes, System.Classes, System.Variants,
  FMX.Types, FMX.Controls, FMX.Forms, FMX.Graphics, FMX.Dialogs, FMX.MediaLibrary,
FMX.Platform, System.Messaging,
  FMX.ListBox, FMX.Edit, FMX.StdCtrls, FMX.Controls.Presentation,
FMX.Objects,FMX.VirtualKeyboard,
  FMX.TabControl, IdBaseComponent, IdComponent, IdTCPConnection, IdTCPClient,
  IdExplicitTLSClientServerBase, IdFTP, System.Actions, FMX.ActnList, FMX.StdActns,
FMX.MediaLibrary.Actions,
  FMX.Filter.Effects, FMX.Effects, FMX.Gestures;

type

  TForm1 = class(TForm)
    TabControl1: TTabControl;
    TabItem1: TTabItem;
    TabItem2: TTabItem;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    Panel4: TPanel;
    Text2: TText;
    Panel5: TPanel;
    Image1: TImage;
    Image4: TImage;
    Image3: TImage;
    Image2: TImage;
    Text3: TText;
    IdFTP1: TIdFTP;
    CornerButton1: TCornerButton;
    ShadowEffect1: TShadowEffect;
```

```
ShadowEffect5: TShadowEffect;
ShadowEffect11: TShadowEffect;
ShadowEffect2: TShadowEffect;
Image6: TImage;
Text7: TText;
Panel6: TPanel;
Panel7: TPanel;
Panel8: TPanel;
Panel9: TPanel;
Panel10: TPanel;
Panel11: TPanel;
Edit1: TEdit;
ShadowEffect13: TShadowEffect;
MonochromeEffect1: TMonochromeEffect;
Text1: TText;
ShadowEffect3: TShadowEffect;
Panel12: TPanel;
Panel13: TPanel;
Panel19: TPanel;
Panel20: TPanel;
Panel21: TPanel;
Edit3: TEdit;
Text5: TText;
Panel14: TPanel;
CornerButton2: TCornerButton;
CornerButton3: TCornerButton;
CornerButton4: TCornerButton;
Panel15: TPanel;
Text4: TText;
Panel16: TPanel;
ShadowEffect4: TShadowEffect;
ShadowEffect10: TShadowEffect;
ShadowEffect12: TShadowEffect;
ShadowEffect6: TShadowEffect;
ShadowEffect7: TShadowEffect;
ShadowEffect8: TShadowEffect;
ShadowEffect9: TShadowEffect;
Image5: TImage;
Rectangle3: TRectangle;
Rectangle4: TRectangle;

procedure FormActivate(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure CornerButton1Click(Sender: TObject);
procedure CornerButton2Click(Sender: TObject);
procedure CornerButton3Click(Sender: TObject);
```

```
    procedure CornerButton4Click(Sender: TObject);

  var
  private

  procedure DoMessageListener(const Sender: TObject; const M: TMessage);
  procedure DoDidFinish(Image: TBitmap);

  public

  end;
  var
  Form1: TForm1;
    const
    path:string = 'storage/emulated/0/Android/Centrolit';

implementation

procedure TForm1.DoDidFinish(Image: TBitmap);
begin

  Unit1.Form1.Image1.Bitmap.Assign(Image);  //Копирование фотографии в битмап.
  Unit1.Form1.Image2.Bitmap.Assign(Image);
  Unit1.Form1.Image3.Bitmap.Assign(Image);
  Unit1.Form1.Image4.Bitmap.Assign(Image);
  Unit1.Form1.Image1.Bitmap.Resize(1440,1920);  //Переопределение разрешение
фотогрфии.
  Unit1.Form1.Image2.Bitmap.Resize(600,800);
  Unit1.Form1.Image3.Bitmap.Resize(480,640);

  // При условии наличия каталога сохраняет 3 изображения разного разрешения в каталог
Centrolit.

      if DirectoryExists('storage/emulated/0/Android/Centrolit') then
    begin

Unit1.Form1.Image1.Bitmap.SaveToFile('storage/emulated/0/Android/Centrolit/'+(Edit1.text)+'_
1440x1920.jpg');

Unit1.Form1.Image2.Bitmap.SaveToFile('storage/emulated/0/Android/Centrolit/'+(Edit1.text)+'_
600x800.jpg');

Unit1.Form1.Image3.Bitmap.SaveToFile('storage/emulated/0/Android/Centrolit/'+(Edit1.text)+'_
480x640.jpg');
    Unit1.Form1.Text2.Text:= 'Успешно выполнено';
    end
```

```
// При отсутствии каталога Centrolit, создаёт его и сохраняет в него 3 изображения разного разрешения.
    else MkDir('storage/emulated/0/Android/Centrolit');
    begin

Unit1.Form1.Image1.Bitmap.SaveToFile('storage/emulated/0/Android/Centrolit/'+(Edit1.text)+'_1440x1920.jpg');
    Unit1.Form1.Image2.Bitmap.SaveToFile('storage/emulated/0/Android/Centrolit/'+(Edit1.text)+'_600x800.jpg');

Unit1.Form1.Image3.Bitmap.SaveToFile('storage/emulated/0/Android/Centrolit/'+(Edit1.text)+'_480x640.jpg');
    Unit1.Form1.Text2.Text:= 'Успешно выполнено';
    Unit1.Form1.Text7.Text:= 'ГЛЗ ЦЕНТРОЛИТ Сотрудник Таб.№'+(Edit1.text)
    end;

end;

// Функция для удаления всех файлов каталога.

procedure RemoveAll(path: string);
var
  sr: TSearchRec;
begin
  if FindFirst(path + '/*.*', faAnyFile, sr) = 0 then
  begin
    repeat
      if sr.Attr and faDirectory = 0 then
      begin
        DeleteFile(path + '/' + sr.name);
      end
      else
      begin
        if pos('.', sr.name) <= 0 then
          RemoveAll(path + '/' + sr.name);
      end;
    until
      FindNext(sr) <> 0;
  end;
  FindClose(sr);
  RemoveDir(PChar(path));
end;

// Обработчки нажатия на клавишу "Сделать фото"
procedure TForm1.CornerButton1Click(Sender: TObject);
```

```
 var
  Service: IFMXCameraService;
  Params: TParamsPhotoQuery;
  Begin
if  Trim(Unit1.Form1.Edit1.Text) = '' then
  begin
   ShowMessage ('Не корректно введён табельный номер.Повторите ввод.');
  end
     // Сэмпл для работы с камерой Android.

     else if TPlatformServices.Current.SupportsPlatformService(IFMXCameraService,
        Service) then
       begin
        Params.Editable := True;
       // Специальные параметры для сохранения изображаения с камеры.
        Params.NeedSaveToAlbum := False;
        Params.RequiredResolution := TSize.Create(5000,5000);
        Params.OnDidFinishTaking := DoDidFinish;
        Service.TakePhoto(CornerButton1,Params);
       end
       else
        ShowMessage('Данное устройство не поддерживает модуль камеры.');
     end;


procedure TForm1.CornerButton2Click(Sender: TObject);

var
dn : integer;
F : TSearchRec;
begin
// Подключение к FTP-серверу.

//Unit1.Form1.idftp1.Host := '194.158.208.111'; //internet
Unit1.Form1.idftp1.Host := 'Base1.glz.local'; //local

//Unit1.Form1.idftp1.Host := '194.158.208.111';
Unit1.Form1.idftp1.Username := 'photoup@glz';
Unit1.Form1.idftp1.Password := '!QAZxsw2#EDC';
Unit1.Form1.idftp1.Port := 2121;
Unit1.Form1.idftp1.Connect;
Unit1.Form1.idftp1.ChangeDir('PHOTO/NEW');


//Перенос всех файлов каталога на FTP-каталог при условии его наличия.
 if  DirectoryExists('storage/emulated/0/Android/Centrolit')then
  begin
```

```
   dn := FindFirst('storage/emulated/0/Android/Centrolit/*.jpg',faDirectory,F);

  while dn = 0 do
  begin
  Unit1.Form1.idftp1.Put('storage/emulated/0/Android/Centrolit/'+F.Name);
  dn:=FindNext(F);
  end;
  ShowMessage ('Успешно скопировано!');
  Unit1.Form1.idftp1.Disconnect;
  end
     else
       ShowMessage ('Каталог пуст.Сделайте фотографию!');
       idftp1.Disconnect;
end;


procedure TForm1.CornerButton3Click(Sender: TObject);
begin

 // Подключение к FTP-серверу.
//Unit1.Form1.idftp1.Host := '194.158.208.111'; //internet
Unit1.Form1.idftp1.Host := 'Base1.glz.local'; //local

//Unit1.Form1.idftp1.Host := '194.158.208.111';
Unit1.Form1.idftp1.Username := 'photoup@glz';
Unit1.Form1.idftp1.Password := '!QAZxsw2#EDC';
Unit1.Form1.idftp1.Port := 2121;
Unit1.Form1.idftp1.Connect;
Unit1.Form1.idftp1.ChangeDir('PHOTO/NEW');

   //Проверка корректного ввода табельного номера.
    begin
        if  Trim(Edit3.Text) = '' then
        begin
        ShowMessage ('Не корректно введён табельный номер.Повторите ввод.');
        Unit1.Form1.idftp1.Disconnect;
        end
   //Копирование изображений по табельному номеру при наличии их в каталоге.
        else if (DirectoryExists('storage/emulated/0/Android/Centrolit')) and
(FileExists('storage/emulated/0/Android/Centrolit/'+(Edit3.text)+'_1440x1920.jpg')) then
        begin

Unit1.Form1.idftp1.put('storage/emulated/0/Android/Centrolit/'+(Edit3.text)+'_1440x1920.jpg');

Unit1.Form1.idftp1.put('storage/emulated/0/Android/Centrolit/'+(Edit3.text)+'_600x800.jpg');

Unit1.Form1.idftp1.put('storage/emulated/0/Android/Centrolit/'+(Edit3.text)+'_480x640.jpg');
```

```
          ShowMessage ('Успешно скопировано!');
          Unit1.Form1.idftp1.Disconnect;
         end

         else
         ShowMessage('Файл с данным табельным номером отсутствует. Либо каталог пуст.');
         Unit1.Form1.idftp1.Disconnect;
    end;
end;

// Процедура очистки каталога
procedure TForm1.CornerButton4Click(Sender: TObject);
begin
RemoveAll(path);
ShowMessage('Каталог успешно очищен!');
end;


procedure TForm1.DoMessageListener(const Sender: TObject; const M: TMessage);
begin
  if M is TMessageDidFinishTakingImageFromLibrary then
  Unit1.Form1.Image4.Bitmap.Assign(TMessageDidFinishTakingImageFromLibrary(M).Value);
end;

{$R *.fmx}
{$R *.NmXhdpiPh.fmx ANDROID}
{$R *.LgXhdpiPh.fmx ANDROID}


procedure TForm1.FormCreate(Sender: TObject);
begin
  unit1.Form1.TabControl1.TabIndex := 0; //Фокус на первом табе.
  Unit1.Form1.Height := screen.Height;  //
  Unit1.Form1.Height := screen.Width;   //Установка размера формы приложения в
соответствии с разрешением экрана.
end;

procedure TForm1.FormShow(Sender: TObject);
begin
ShowMessage ( 'Перед работой установите параметр съёмки камеры в формате 4x3!')
end;

procedure TForm1.FormActivate(Sender: TObject);
begin
Unit1.Form1.Edit1.SetFocus;
end;
```

```
end.
```

ПРИЛОЖЕНИЕ Б

Инструкция инженера-программиста

Инженер-программист должен знать:

– руководящие и нормативные материалы, регламентирующие методы разработки алгоритмов и программ и использования вычислительной техники при обработке информации, основные принципы структурного программирования;

– виды программного обеспечения;

– технико-эксплуатационные характеристики, конструктивные особенности, назначение и режимы работы ЭВМ, правила ее технической эксплуатации;

– технологию автоматической обработки информации;

– виды технических носителей информации;

– методы классификации и кодирования информации;

– формализованные языки программирования;

– действующие стандарты, системы счислений, шифров и кодов;

– порядок оформления технической документации;

– передовой отечественный и зарубежный опыт программирования и использования вычислительной техники;

– основы экономики, организации производства, труда и управления;

– основы трудового законодательства;

– правила и нормы охраны труда.

Должностные обязанности:

– на основе анализа математических моделей и алгоритмов решения экономических и других задач разрабатывает программы, обеспечивающие возможность выполнения алгоритма и соответственно поставленной задачи средствами вычислительной техники, проводит их тестирование и отладку;

– разрабатывает технологию решения задачи по всем этапам обработки информации;

– осуществляет выбор языка программирования для описания алгоритмов и структур данных;

– определяет информацию, подлежащую обработке средствами вычислительной техники, ее объемы, структуру, макеты и схемы ввода, обработки, хранения и вывода, методы ее контроля;

– выполняет работу по подготовке программ к отладке и проводит отладку;

– определяет объем и содержание данных контрольных примеров, обеспечивающих наиболее полную проверку соответствия программ их функциональному назначению;

– осуществляет запуск отлаженных программ и ввод исходных данных, определяемых условиями поставленных задач;

– проводит корректировку разработанной программы на основе анализа выходных данных;

– Информирует руководство об имеющихся недостатках в работе предприятия, принимаемых мерах по их ликвидации;

– разрабатывает инструкции по работе с программами, оформляет необходимую техническую документацию;

– определяет возможность использования готовых программных продуктов;

– осуществляет сопровождение внедренных программ и программных средств;

– разрабатывает и внедряет системы автоматической проверки правильности программ, типовые и стандартные программные средства, составляет технологию обработки информации;

– выполняет работу по унификации и типизации вычислительных процессов;

– принимает участие в создании каталогов и картотек стандартных программ, в разработке форм документов, подлежащих машинной обработке, в проектировании программ, позволяющих расширить область применения вычислительной техники.

ПРИЛОЖЕНИЕ В

Требования по охране труда и технике безопасности

В области охраны труда инженер-программист обязан:

– участвовать в реализации политики и целей в области охраны труда в пределах своих полномочий;

– соблюдать требования по охране труда и пожарной безопасности, а также правила поведения на территории организации, в аудиториях, вспомогательных и бытовых помещениях;

– использовать и правильно применять средства индивидуальной защиты;

– проходить в установленном законодательством порядке медицинские осмотры, обучение, инструктаж и проверку знаний по вопросам охраны труда;

– заботиться о личной безопасности и личном здоровье, а также о безопасности окружающих в процессе выполнения работ либо во время нахождения на территории университета;

– немедленно сообщать непосредственному руководителю о любой ситуации угрожающей жизни или здоровью работающих и окружающих, несчастном случае, произошедшем на производстве, оказывать содействие непосредственному руководителю в принятии мер по оказанию необходимой помощи потерпевшим и доставке их в организацию здравоохранения;

– выполнять нормы и обязательства по охране труда, предусмотренные коллективным договором, соглашением, трудовым договором, правилами внутреннего трудового распорядка, должностными обязанностями;

– в случае отсутствия средств индивидуальной защиты немедленно уведомлять об этом непосредственного руководителя либо иного уполномоченного должностного лица нанимателя;

– оказывать содействие и сотрудничать с непосредственным руководителем в деле обеспечения здоровых и безопасных условий труда, немедленно извещать своего непосредственного руководителя или иного уполномоченного должностного лица нанимателя о неисправности

оборудования, приспособлений, средств защиты, об ухудшении состояния своего здоровья;

– поддерживать свое рабочее место, оборудование и приспособления в исправном состоянии, порядке и чистоте;