Министерство науки и высшего образования Российской Федерации

Министерство образования и науки Кыргызской Республики

Кыргызско-Российский Славянский университет

Естественно-Технический Факультет

**Отчет по курсовой работе**

**Дисциплина : Разработка Корпоративных приложений**

**Тема: Бюро по трудоустройству**

**Выполнил:**

Ивженко Кирилл Епи 2-20

**Проверил(а):**

Турчанова Т.Г.

Дата:\_\_\_\_\_\_\_\_\_\_\_\_

Бишкек 2023

Оглавление

[Введение 3](#_Toc136912731)

[Содержательную постановку задачи 4](#_Toc136912732)

[Обоснование выбора субд и языка программирования 5](#_Toc136912733)

[Описание структуры разработанной базы данных 6](#_Toc136912734)

[Описание программы (руководство программиста) 7](#_Toc136912735)

[Краткое руководство пользователя 11](#_Toc136912736)

[Заключение 17](#_Toc136912737)

[Источники. 17](#_Toc136912738)

# Введение

В современном обществе вопрос трудоустройства и поиска работы является одной из наиболее актуальных и сложных задач. Со временем рынок труда стал все более конкурентным и динамичным, требуя от соискателей знаний и навыков, способных отразить современные требования работодателей. В связи с этим появляется необходимость в организации профессиональной поддержки и помощи для тех, кто ищет работу или желает сменить свою текущую занятость.

Бюро по трудоустройству является одним из институтов, созданных для решения проблемы соответствия требованиям рынка труда и потенциальных кандидатов на вакансии. Это организация, предоставляющая разнообразные услуги, направленные на содействие трудоустройству и профессиональному развитию работников.

Целью данной курсовой работы является анализ и оценка деятельности бюро по трудоустройству в контексте современных вызовов и требований рынка труда. В работе будут рассмотрены основные функции, задачи и услуги, предоставляемые бюро по трудоустройству, а также оценены их эффективность и соответствие потребностям работодателей и соискателей.

Для достижения данной цели будет проведен обзор существующих научных источников, а также анализ статистических данных, связанных с работой бюро по трудоустройству. Будут рассмотрены преимущества и недостатки деятельности бюро, а также предложены рекомендации по улучшению и оптимизации их работы.

Результаты и выводы данной работы могут быть полезны для организаций, занимающихся трудоустройством, а также для соискателей и работодателей, которые заинтересованы в повышении эффективности процесса подбора персонала и успешном трудоустройстве.

# Содержательную постановку задачи

Целью данной курсовой работы является разработка программного решения для бюро по трудоустройству, которое будет автоматизировать процессы поиска работы и управления кандидатами. Для достижения данной цели ставятся следующие задачи:

1. Разработать базу данных, которая будет хранить информацию о доступных вакансиях, кандидатах и работодателях. База данных должна быть гибкой и эффективной для быстрого поиска и обновления информации.
2. Создать пользовательский интерфейс, который позволит работникам бюро легко добавлять новые вакансии, редактировать информацию о кандидатах, а также просматривать и обрабатывать заявки на работу.
3. Реализовать систему рекомендаций, основанную на анализе соответствия кандидатов и вакансий. Это позволит более точно подбирать наиболее подходящих кандидатов для каждой конкретной вакансии и увеличить шансы на успешное трудоустройство.
4. Включить функциональность автоматизированной отправки уведомлений кандидатам о новых вакансиях, изменениях в статусе их заявок и других важных событиях. Это поможет поддерживать активную коммуникацию с кандидатами и повысить их участие в процессе трудоустройства.
5. Обеспечить безопасность данных, реализовав соответствующие механизмы шифрования и авторизации доступа к информации. Это гарантирует сохранность и конфиденциальность личных данных кандидатов и работодателей.

Результатом данной работы будет функциональное программное решение, способное упростить и оптимизировать процессы трудоустройства в бюро, а также повысить эффективность подбора персонала. Разработанное программное обеспечение будет полезным для бюро по трудоустройству, которые смогут автоматизировать свои операции и повысить качество предоставляемых услуг.

# Обоснование выбора субд и языка программирования

В качестве языка программирования был выбран язык c#, так как он предоставляется большой спектр возможностей для реализации данной задачи, в пример можно взять такую технологию как EntityFramework, которая представляет собой объектно-ориентированную, легковесную и расширяемую технологию от компании Microsoft для доступа к данным . EF Core является ORM-инструментом (object-relational mapping - отображения данных на реальные объекты). То есть EF Core позволяет работать с базами данных, но представляет собой более высокий уровень абстракции.

В качестве технологию программирования была выбрана технология ORM. ORM (англ. Object-Relational Mapping) – это технология программирования, связывающая СУБД с объектно-ориентированным языком программирования, создавая «виртуальную объектную базу данных». Я выбрал подход «Db first», так ка посчитал, что так будет лучше.

В качестве Паттерна проектирования был выбран Паттерн MVC

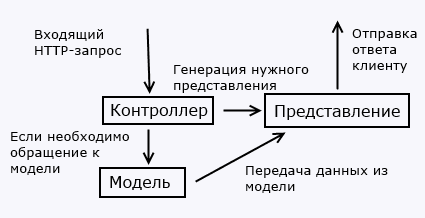
На следующей схеме показаны три основных компонента и существующие между ними связи.

Рисунок 1 Паттерн MVC

В качестве Субд была выбран SQL Server

# Описание структуры разработанной базы данных

Рисунок 2 Структура БД Бюро по трудоустройству

В основе каждого бюро по трудоустройству будет лежать примерно такая структура, где у нас есть Соискатель и Работодатель, который открывает новые вакансии.

# Описание программы (руководство программиста)

Функциональная схема данной программы приведена ниже.

Вход

Регистрация

Панель Пользователя

Отчет всех пользователей

Панель Админа

Добавление анкеты работодателя

Добавление анкеты соискателя

Вакансии и соискатели

Предложения мне как соискателю, принятие, удаление

Мои вакансии, принятие соискателей, удаление вакансий

Образование, Должности

Мой отчет

Основа всего Соискатель и Работодатель, поэтому все будете подстраиваться под них, Необходимо создать возможности их добавления, поиска нужных вакансий или соискателей, подбор вакансий или соискателей, для конкретной заявки, Ниже будут приведены примеры кода, добавления, удаления, поиска.

public void AddApplicant(string Fio, string Gender, DateTime dateofbirth, string Phonenumber, string education, string Pos, bool drivelic, string salary, string city, bool Pc\_kn, string Languages, string WorkExp, string Schedule)

{

try

{

var Appl = db.Applicants.Any(p => p.Applicantid == userid);

if (Appl)

{

if (Fio == "") throw new Exception();

if (Phonenumber == "") throw new Exception();

var Positionindbup = db.Positions.Where(p => p.Position1 == Pos).First().Positionid;

var educindbup = db.Educations.Where(p => p.Education1 == education).First().Educationid;

int salarytodbup;

bool salaryparseup = int.TryParse(salary, out salarytodbup);

if (salary == "" || salaryparseup == false || salarytodbup <= 0 || int.Parse(WorkExp) < 0) throw new Exception();

string[] Langup = Languages.Split(',');

var aplupdate = db.Applicants.Where(p => p.Applicantid == userid).First();

aplupdate.Languages = Languages;

aplupdate.Fio = Fio;

aplupdate.Gender = Gender;

aplupdate.DateofBirth = dateofbirth;

aplupdate.PhoneNumber = Phonenumber;

aplupdate.Education = educindbup;

aplupdate.Position = Positionindbup;

aplupdate.DriverLicense = drivelic;

aplupdate.Salary = salarytodbup;

aplupdate.City = city;

aplupdate.PcKnowledge = Pc\_kn;

aplupdate.WorkExperience = WorkExp;

aplupdate.Schedule = Schedule;

db.SaveChanges();

db.ApplicantLanguages.RemoveRange(db.ApplicantLanguages.Where(p => p.Applicant == userid).ToList());

db.SaveChanges();

foreach (var lang in Langup)

{

var islangofAppl = db.ApplicantLanguages.Any(p => p.Applicant == userid && p.Language == lang);

if (!islangofAppl)

{

db.ApplicantLanguages.Add(new ApplicantLanguage

{

Applicant = userid,

Language = lang

});

db.SaveChanges();

}

}

MessageBox.Show("Анкета изменена");

return;

}

if (Fio == "") throw new Exception();

if (Phonenumber == "") throw new Exception();

var Positionindb = db.Positions.Where(p => p.Position1 == Pos).First().Positionid;

var educindb = db.Educations.Where(p => p.Education1 == education).First().Educationid;

int salarytodb;

bool salaryparse = int.TryParse(salary, out salarytodb);

if (salary == "" || salaryparse == false) throw new Exception();

string[] Lang = Languages.Split(',');

db.Applicants.Add(new Applicant

{

Applicantid = userid,

Fio = Fio,

PlacementDate = DateTime.Now,

DateofBirth = dateofbirth,

Salary = int.Parse(salary),

Schedule = Schedule,

DriverLicense = drivelic,

Languages = Languages,

Gender = Gender,

Position = Positionindb,

Education = educindb,

City = city,

PhoneNumber = Phonenumber,

WorkExperience = WorkExp,

PcKnowledge = Pc\_kn

});

db.SaveChanges();

foreach (var lang in Lang)

{

var islangofAppl = db.ApplicantLanguages.Any(p => p.Applicant == userid && p.Language == lang);

if (!islangofAppl)

{

db.ApplicantLanguages.Add(new ApplicantLanguage

{

Applicant = userid,

Language = lang

});

db.SaveChanges();

}

}

MessageBox.Show("Ваша анкета добавлена");

}

catch

{

MessageBox.Show("Проверьте, все ли поля заполнены", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error, MessageBoxDefaultButton.Button1, MessageBoxOptions.DefaultDesktopOnly);

}

}

public void Deleteappl(int userid)

{

var appl = db.Applicants.Where(p => p.Applicantid == userid).First();

db.Applicants.Remove(appl);

db.SaveChanges();

addApplicant.Close();

}

public void Serch(string Gender, string Employer, string education, string Pos, bool drivelic, int salary, string city, bool Pc\_kn, DataGridViewSelectedRowCollection selectedLanguages, int WorkExp, string Schedule, DataGridView vacs,int userid)

{

var educationParam = new SqlParameter("@educationparam", education);

var positionParam = new SqlParameter("@position", Pos);

var genderParam = new SqlParameter("@gender", Gender);

var driverLicenseParam = new SqlParameter("@driverLicense", drivelic);

var salaryParam = new SqlParameter("@salary", salary);

var cityParam = new SqlParameter("@city", "%" + city + "%");

var pcKnowledgeParam = new SqlParameter("@pcKnowledge", Pc\_kn);

var workExperienceParam = new SqlParameter("@workExperience", WorkExp);

var scheduleParam = new SqlParameter("@schedule", Schedule);

var companyParam = new SqlParameter("@company", Employer);

var userparam = new SqlParameter("@user", userid);

var selectedLanguageNames = selectedLanguages.Cast<DataGridViewRow>()

.Select(row => row.Cells["Language"].Value.ToString())

.ToList();

var languagesParam = new SqlParameter("@languages", string.Join(" OR ", selectedLanguageNames.Select(lang => $"CONTAINS(v.Languages, '{lang}')")));

var result = db.Vacancies

.FromSqlRaw("SELECT v.\* " +

"FROM Vacancies v " +

"INNER JOIN Employers e ON v.Employer = e.Empoyerid " +

"INNER JOIN Education educ ON v.Education = educ.educationid " +

"INNER JOIN Positions pos ON v.Position = pos.positionid " +

"INNER JOIN Vacancy\_Languages vl ON v.Vacancyid = vl.Vacancy " +

"WHERE educ.Education LIKE @educationparam " +

"AND pos.Position LIKE @position " +

"AND v.Openness = 1 " +

"AND v.gender LIKE @gender " +

"AND v.Driver\_license = @driverLicense " +

"AND v.Salary >= @salary " +

"AND v.City LIKE @city " +

"AND v.PC\_knowledge = @pcKnowledge " +

"AND v.work\_experience <= @workExperience " +

"AND v.Schedule LIKE @schedule " +

"AND e.Userid != @user " +

"AND e.Company LIKE @company",

educationParam, positionParam, genderParam, driverLicenseParam, salaryParam, cityParam, pcKnowledgeParam, workExperienceParam, scheduleParam, companyParam, userparam).ToList();

var filteredVacancies = result.Where(vacancy => selectedLanguageNames.All(language => vacancy.Languages.Contains(language))).ToList();

var vacsearch = (from vac in filteredVacancies

join emp in db.Employers on vac.Employer equals emp.Empoyerid

join educ in db.Educations on vac.Education equals educ.Educationid

join pos in db.Positions on vac.Position equals pos.Positionid

select new

{

Company = emp.Company,

НомерТелефона = emp.PhoneNumber,

Vacancyid = vac.Vacancyid,

Salary = vac.Salary,

График = vac.Schedule,

ВодПрава = vac.DriverLicense,

Языки = vac.Languages,

Образование = educ.Education1,

Пол = vac.Gender,

Должность = pos.Position1,

Город = vac.City,

Опыт = vac.WorkExperience,

ЗнаниПк = vac.PcKnowledge

}).ToList();

vacs.DataSource = vacsearch;

if(vacsearch.Count() != 0) vacs.Columns["Vacancyid"].Visible = false;

}

Поиск вообще довольно сложный процесс, так как у нас есть множество критериев, и какие то мы можем выбрать, а какие – то нет.

Также должно быть предоставление отчета, он может быть предоставлен за весь период, но может и за конкретный.

public void GetEmpouersforRequestDate(int userid, DataGridView employers,DateTimePicker f, DateTimePicker s)

{

var mydeals = (from mydeal in db.DealApplicants

join deal in db.Deals on mydeal.Dealid equals deal.Dealid

join vac in db.Vacancies on deal.Vacancy equals vac.Vacancyid

join emp in db.Employers on vac.Employer equals emp.Empoyerid

join educ in db.Educations on vac.Education equals educ.Educationid

join pos in db.Positions on vac.Position equals pos.Positionid

where vac.Openness == false && mydeal.Applicantid == userid && deal.Dateofpreparation >= f.Value && deal.Dateofpreparation <= s.Value

select new

{

Компания = emp.Company,

Зарплата = vac.Salary,

Коммисионные = deal.Commission,

образование = educ.Education1,

Должность = pos.Position1,

ДатаРазмещение = vac.PlacementDate

}).ToList();

employers.DataSource = mydeals;

}

# Краткое руководство пользователя

Прежде всего нужно войти в приложение под своим аккаунтом, если он есть, если нет, то необходимо создать его, ниже приведены рисунки, иллюстрирующие данные действия.

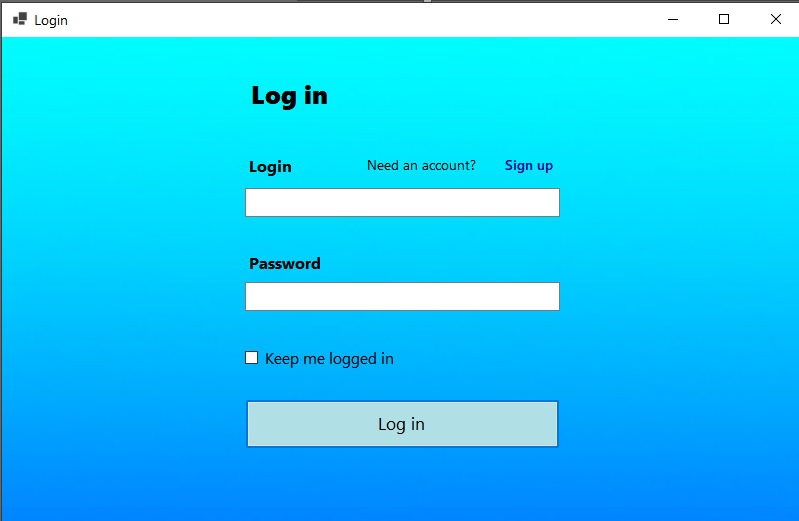


Рисунок 3 Вход

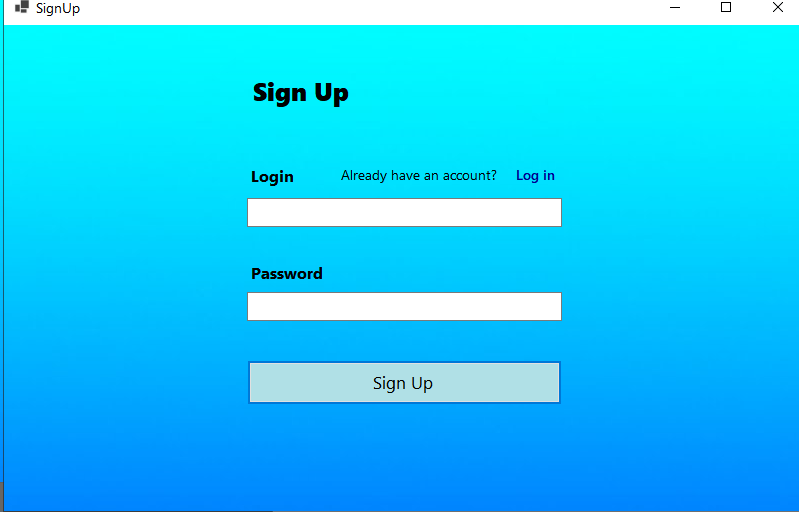
Для регистрации нужно нажать на Sign up, и откроется данное окно

Рисунок 4 Регистрация

Если мы вошли, как админ, то открывается такое окно



Рисунок 5 Панель Админа

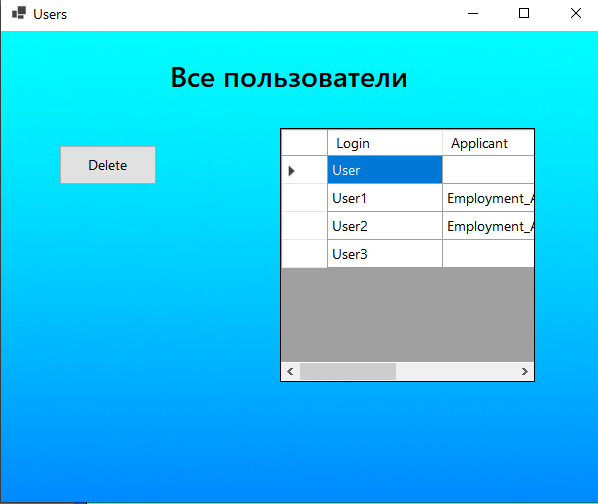
Здесь мы можем видеть, вакансии, работодателей и соискателй, для этого нужно нажимать на соответсвующие кнопки. Также есть возможность их удалять. При нажатии на кнопку «Users» мы видим всех пользователей.

Рисунок 6 Окно пользователей

Если мы хотим добавить Вид образования или должность, нажимаем на соответсвующую кнопку

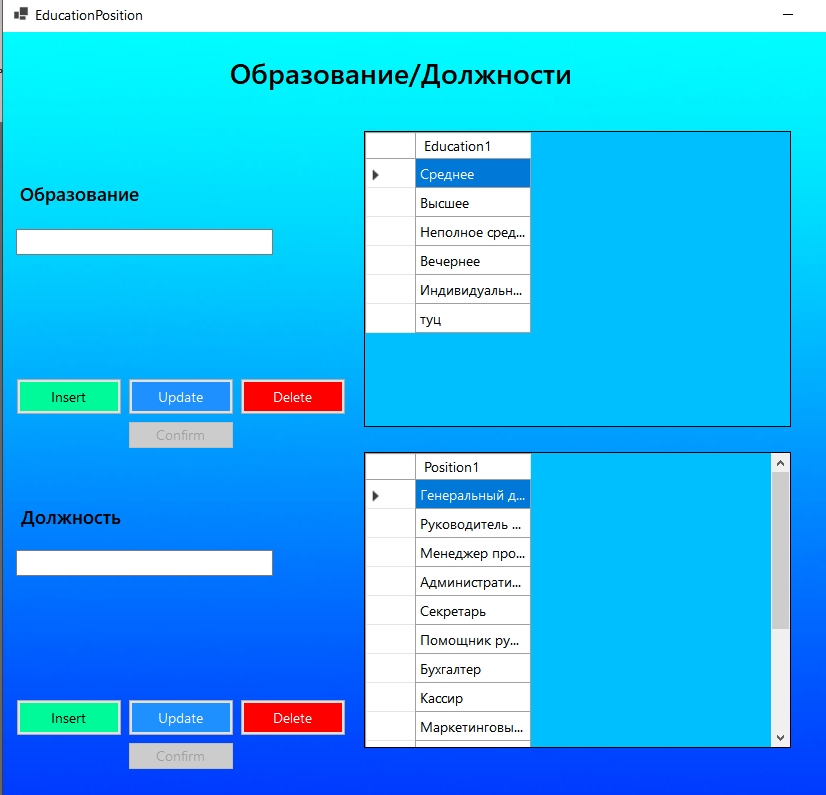


Рисунок 7 Окно Видов образований и должностей

Также можно осуществлять поиск.

Рисунок 8 Главное меню

Если мы вошли как обычный пользователь, мы видим такое окно

И здесь, мы можем выбрать, что мы хотим сделать, например мы можем добавить анкету соискателя

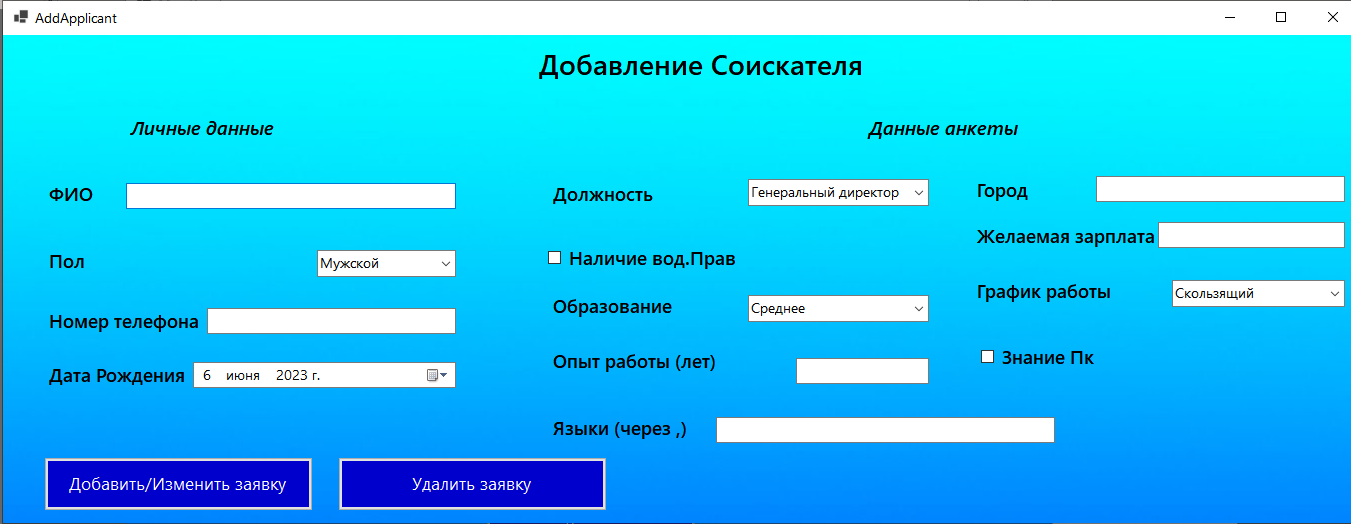


Рисунок 9 Окно добавления анкеты соискателя

Можем получить отчет.

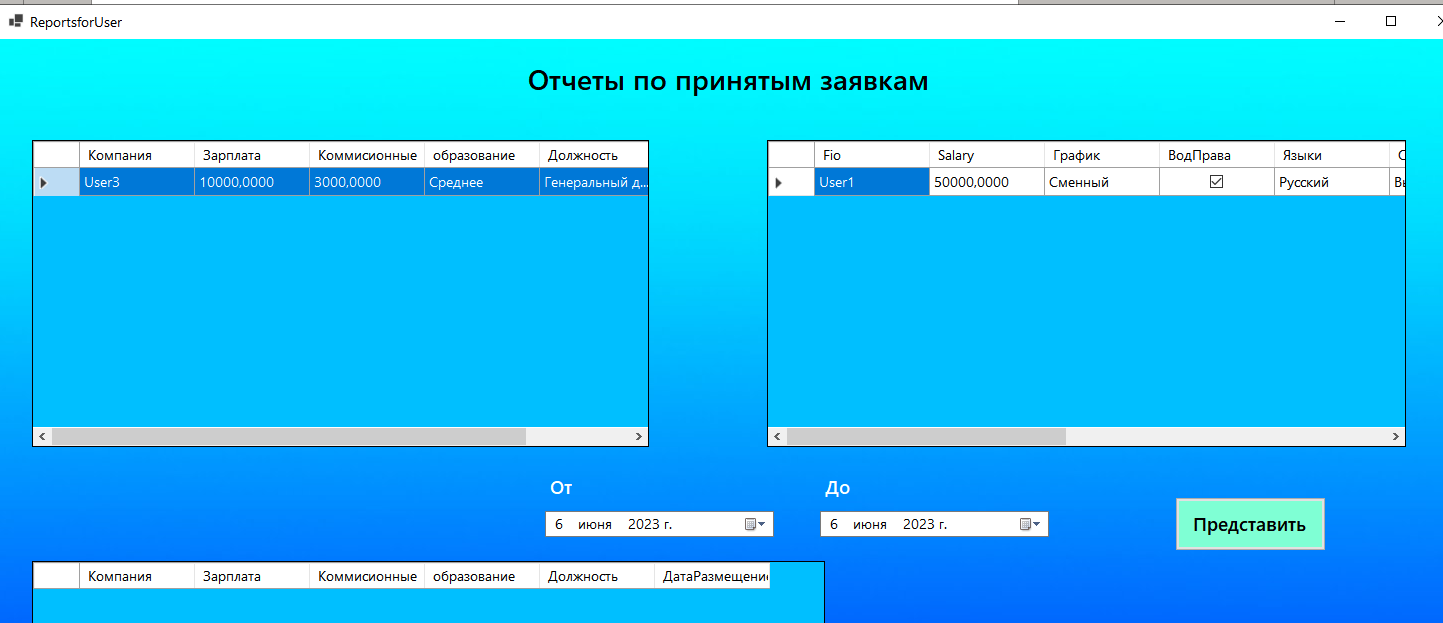


Рисунок 10 Окно отчетов

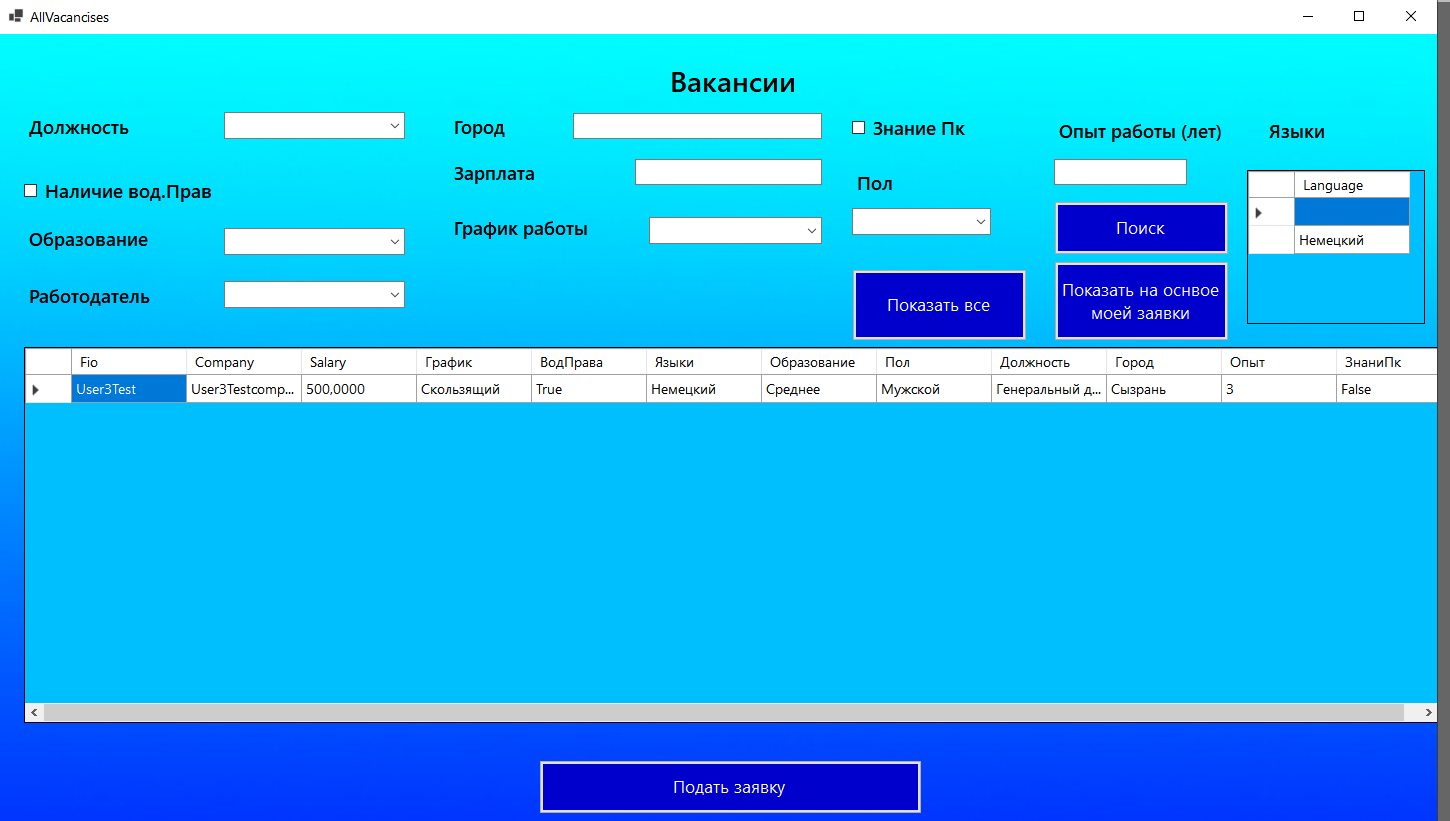
Смотреть свои вакансии и не только, а также можем подавать наши заявки, либо соискателям, либо работодателям.

Рисунок 11 Окно вакансий

Нажав на кнопку «Заявления соискателй» мы можем видеть анкеты соискателей, подавнные для конкретной вакансии.

Нажав на кноку «Предложения работодателей», мы видим предложенные нам вакансии от работодателй. После всех действий можно и выйти из приложения, нажав на кнопку Выйти.

# Заключение

В заключение, данное исследование привело к нескольким важным выводам и результатам. Проведенный анализ данных показал, что реализованная система управления проектами значительно повысила эффективность и точность планирования проектов в организации. Было достигнуто снижение времени выполнения задач, оптимизация ресурсов и повышение качества конечных продуктов.

Проведенные эксперименты и тестирование системы подтвердили ее надежность и стабильность работы, а также выявили некоторые потенциальные области для дальнейших улучшений, такие как повышение масштабируемости и добавление дополнительных функциональностей.

В целом, цели и задачи, поставленные перед нами в начале работы, были успешно достигнуты. Организация получила значимые преимущества от внедрения новой системы управления проектами, включая улучшение взаимодействия команды, сокращение времени и затрат на проекты, а также увеличение уровня удовлетворенности клиентов.

# Источники.

1. https://dspace.tltsu.ru/jspui/bitstream/123456789/9691/1/%D0%9F%D0%B0%D0%B2%D0%BB%D0%BE%D0%B2%20%D0%94.%D0%A1.\_%D0%9F%D0%98%D0%B1%D0%B4-1402%D0%B0.pdf