Федеральное агентство по образованию РФ

ГОУ ВПО Нижегородский государственный университет им. Н.И. Лобачевского

Факультет Вычислительной математики и кибернетики

Кафедра Математического обеспечения ЭВМ

УЧЕБНЫЙ КУРС

**«Объектно-ориентированный анализ и проектирование»**

для подготовки по направлению «Информационные технологии»

КОНЦЕПЦИЯ ПРОЕКТА

Нижний Новгород

2015

###### Содержание[1]

1. Необходимость проекта

1.1. Обоснование необходимости

1.2. Видение проекта

1.3. Анализ выгод

2. Концепция решения

2.1. Цели и Задачи

2.2. Предположения и Ограничения

2.3. Анализ использования

2.3.1. Пользователи

2.3.2. Сценарии использования

2.4. Требования

2.4.1. Требования пользователей

2.4.2. Системные требования

3. Рамки

3.1. Функциональность решения

3.2. За рамками решения

3.3. Критерии одобрения решения

4. Стратегии дизайна решения

4.1. Стратегия архитектурного дизайна

4.2. Стратегия технологического дизайна

# 1. Необходимость проекта

## 1.1. Обоснование необходимости

В настоящее время у многих разработчиков возникает проблема отсутствия удобного и надежного сервиса для поиска реализации нужного алгоритма. Отсутствие такого инструмента приводит к тому, что разработчики тратят много ресурсов на разработку нужного решения, хотя могли бы потратить гораздо меньше сил и средств, приобретя готовую реализацию.

С другой стороны, многие разработчики имеют свои собственные реализации популярных и востребованных алгоритмов, но они хранятся у них без дела. При наличии сервиса для обмена алгоритмами, такие разработчики смогут не только поделиться своими решениями с другими разработчиками, но и зарабатывать, продавая свои реализации.

С подобными проблемами программисты сталкиваются довольно часто, поэтому возникло желание со стороны различных разработчиков найти решение, которое упростило бы обмен решениями и стало бы удобным для всех. Программный продукт, решающий вышеописанные проблемы, планируется быть реализованным в дальнейшей работе.

## 1.2. Видение проекта

Наше видение: удобный и эффективный WEB-сервис поиска и покупки алгоритмов. Многие разработчики нуждаются в простых решениях для написания и развития сложных систем, однако не всегда находятся ресурсы для реализации трудоёмких алгоритмов. В то же время другие разработчики уже имеют готовые решения и готовы поделиться ими за определенную плату.

Наша система является связующей компонентой между ними и даёт возможность: обмениваться реализациями алгоритмов (свободно/платно), проводить проверку корректности (компиляция, запуск тестов).

Успех проекта определяется выполнением всех поставленных задач в двухмесячный срок.

## 1.3. Анализ выгод

В результате внедрения и использования системы, пользователи получат возможность продавать и распространять свои реализации алгоритмов, а также применять решения других пользователей в своих программных продуктах.

Основной выгодой для заказчика будет являться получение процента с операций по покупке/продаже алгоритмов. Также появится возможность покупать алгоритмы у сторонних разработчиков и использовать их.

Выгода исполнителя - приобретение опыта в разработке web-сервисов и успешное выполнение проекта.

# 2. Концепция решения

## 2.1. Цели и Задачи

1. Пользователи смогут размещать реализации своих алгоритмов.

a. Сделать проверку на компилируемость.

b. Предоставить тестирование алгоритмов, размещаемых пользователем.

i. Задать входные/выходные параметры алгоритмов

ii. Задать тестовый набор данных

c. Предоставить возможность задания уровня доступа (платный/свободный).

2. Пользователи смогут осуществить поиск нужного алгоритма из списка предложенных.

a. Реализовать поиск алгоритмов по ключевым словам.

3. Появится возможность покупки алгоритмов.

a. Осуществить возможность покупки необходимой реализации алгоритма с помощью банковской карты.

b. Добавить возможность тестирования алгоритма на пользовательских данных перед покупкой.

4. Система будет предоставлять алгоритмы на различных языках программирования: С/С++, C#, Pascal.

a. Предоставить возможность расширения системы для поддержки нескольких языков программирования.

## 2.2. Предположения и Ограничения

● Поддержка браузера Google Chrome.

● Алгоритмы, добавляемые в систему должны работать не более 10 минут на предоставляемых тестовых наборах.

● Объем тестовых данных не должен превышать 1 мегабайта

● Исходный код алгоритма должен быть помещен в единственный файл

● Привязка к реальным платёжным системам реализована не будет

## 2.3. Анализ использования

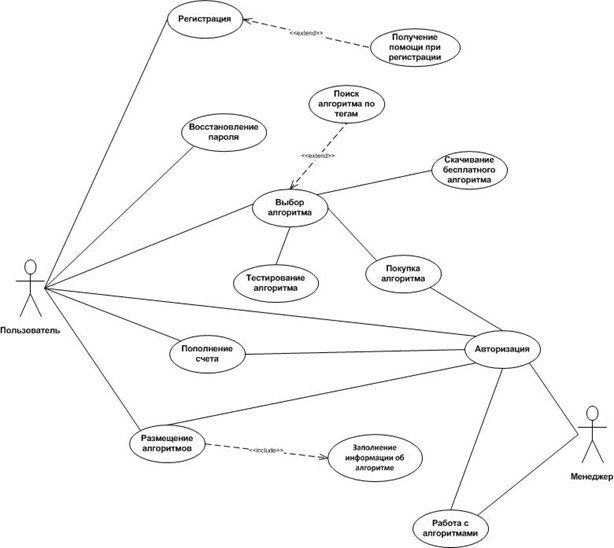
### 2.3.1. Пользователи

● Разработчики алгоритмов

● Разработчики, которые хотят использовать имеющиеся алгоритмы в своих решениях

● Менеджер как администратор системы

### 2.3.2. Сценарии использования [Koля AR: добавить работу с данными, пользователя отнаследовать о т двух ублюдков + в UML интерфейс обозначаются другой стрелкой]



## 2.4. Требования

### 2.4.1. Требования пользователей

1. Регистрация и авторизация пользователей

2. Пользовательские данные защищены от несанкционированного доступа

3. Возможность загружать исходный код алгоритмов

4. Возможность задавать тестовые данные

5. Возможность протестировать работу алгоритма на заданных данных

6. Возможность разграничивать уровни доступа к алгоритму (платный/свободный)

7. Возможность задавать теги для поиска алгоритмов

8. Возможность скачать/приобрести выбранный алгоритм

9. Возможность менеджера модерировать данные системы

### 2.4.2. Системные требования

Сформулируйте требования к решению с точки зрения среды, в которой оно должно будет функционировать после внедрения.

1. Сервер работает под управлением ОС Windows

2. C/C++ компилятор: MSVC 2010

3. C# компилятор: csc.exe

4. FreePascal компилятор: (ppc386 или ppcrossx64)

5. HDD, RAM, CPU Cores

6.

7.

# 3. Рамки

## 3.1. Функциональность решения

1. Доступ к системе осуществляется через браузер по заданному адресу

2. Регистрация и авторизация пользователей по паре “адрес электронной почты - пароль”

3. Личный счёт пользователя, с помощью которого можно покупать алгоритмы

4. Восстановление пароля

5. Поиск по ключевым словам среди всех алгоритмов в системе

6. Запуск выбранного алгоритма на авторских или загруженных пользователем данных

7. Доступ к исходному коду выбранного алгоритма, распространяемого платно, имеется только после оплаты

8. Добавление новой реализации алгоритма

9. Задание ключевых слов для своего алгоритма

10. Задание тестовых данных для своего алгоритма

11. Управление доступом к своему алгоритму

12. Модерация данных системы менеджером

## 3.2. За рамками решения

Поддержка реальных платёжных систем для покупки и продажи алгоритмов. Эта функциональность остаётся за рамками решения по следующим причинам: на данном этапе это студенческий проект, не подразумевающий реального использования, следовательно не рационально тратить ресурсы на поддержку данной функциональности.

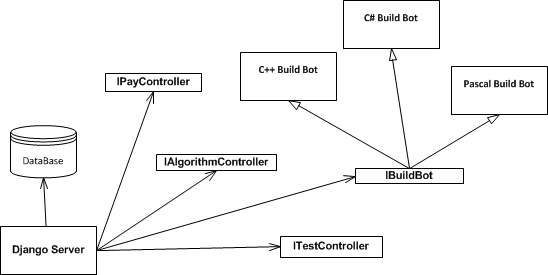
## 3.3. Критерии одобрения решения

1. Реализованы требования пользователей (2.4.1)

2. Успешно пройдено тестирование, описанное в документе “Спецификации и сценарии тестов”

# 4. Стратегии дизайна решения

## 4.1. Стратегия архитектурного дизайна



## 4.2. Стратегия технологического дизайна

Система будет реализована на языках Python, JavaScript с использованием фреймворка Django и базы данных SQLite. Эти средства выбраны из-за простоты разработки веб-сервисов с помощью Django и мощных возможностей языка Python для автоматизации.

[1] В документе использованы материалы белых книг (white papers) “MSF Process Model”, “MSF Risk Management Discipline”, “MSF Team Model” (<http://www.microsoft.com/msf>), их переводов “Модель процессов MSF”, “Дисциплина управления рисками MSF”, “Модель проектной группы MSF” выполненных в 2003 году корпораций eLine Software ([http://www.elinesoftware.com](http://www.elinesoftware.com/)), а также официальных курсов Microsoft 2710B и 1846A.