Министерство образования и науки Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(**ВлГУ)**

Кафедра "Информационных систем и программной инженерии"

Отчет

по дисциплине

«Распределительные программные системы»

Этап № 2

по Курсовому проекту

на тему

“Разработка программной системы

«AroundMe»”

Выполнили:

ст. гр. ПРИ-116

Никольский К.В

Мартынов Г.И.

Принял:

доц. Тимофеев А.А.

Владимир, 2018

1 ПОСТАНОВКА ЦЕЛИ И ЗАДАЧ

1.1 Цель

1.1.1 Получить автоматизированную систему взаимодействия пользователей, организации ими событий, участие в них, создание заданий и предложение жилья. Добавить возможность создания и редактирования собственного социального паспорта пользователя. Реализовать в виде мобильного приложения.

1.2 Задачи

1.2.1 Для достижения цели, необходимо решить следующие задачи:

− изучение особенностей похожих приложений (аналогов) и подбор языков программирования для его реализации;

− анализ возможных функций, которые должны быть в реализуемой системе;

− разработка диаграмм (прецедентов, классов, состояний, навигации, автоматизируемого бизнес-процесса) программной системы;

− разработка моделей (структурной, информационной, данных) программной системы;

− выбор алгоритмов программирования с учетом их точности и спецификой предметной области;

− разработка программного обеспечения;

− анализ полученных результатов работы ПО.

**1.3 Список аналогов**

− Соц. сеть Вконтакте (приложение, сайт, события);

− Facebook (приложение, сайт, события);

− Avito (Работа);

− Airbnb (Жилье);

− booking.com‎ (Жилье);

− Google Карты.

**1.4 Команда, роли в команде**

1.4.1 Команда состоит из 2 студентов группы ПРИ-116:

* Никольский К. В. (Автор идеи, редактор, программист, дизайнер, тестировщик);
* Мартынов Г. И. (Автор идеи, руководитель проекта, разработчик, аналитик, программист, дизайнер, тестировщик);
* разработчик интерфейса.

1.5 Календарный план проекта

1.5.1 Данные для построения диаграммы Гантта представлены в таблице 1.

Таблица 1 – Данные для построения диаграммы Гантта

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Этап проекта | | Начало | Длительность | Задержка | Конец |
| Никольский | Мартынов |
| 1.Название проекта 2.Перечень функциональных требований к системе 3.Команда, роли в команде | | 24.09.2018 | 4 | -3 | 28.09.2018 |
|  | Диаграмма Гантта | 25.09.2018 | 1 | 12 | 25.09.2018 |
| Обзор актуальных технологий | 1.Описание предметной области  2. Словарь предметной области  3.Диаграмма прецедентов | 06.10.2018 | 2 | 6 | 07.10.2018 |
|  | Описание прецедентов | 13.10.2018 | 1 | -6 | 13.10.2018 |
| Анализ аналогов | 1. Пользователи системы и роли 2. Диаграмма классов | 07.10.2018 | 14 | 0 | 21.10.2018 |
|  | Список аналогов | 21.10.2018 | 1 | 0 | 21.10.2018 |
| Список аналогов | Диаграмма состояний | 21.10.2018 | 2 | 5 | 22.10.2018 |
| 1. Перечень экранных форм 2. Диаграмма навигации |  | 27.10.2018 | 2 | 0 | 28.10.2018 |
| Эскизы экранных форм |  | 28.10.2018 | 7 | -1 | 03.11.2018 |
|  | Диаграмма автоматизируемого бизнес-процесса | 02.11.2018 | 2 | 0 | 03.11.2018 |
|  | Нефункциональные требования к системе | 03.11.2018 | 1 | 0 | 03.11.2018 |
| Разработка интерфейса |  | 03.11.2018 | 3 | 12 | 05.11.2018 |
| Презентация идеи проекта | | 17.11.2018 | 2 | 0 | 18.11.2018 |

1.5.2 Диаграммы Гантта представлены на рисунке 1, 2.

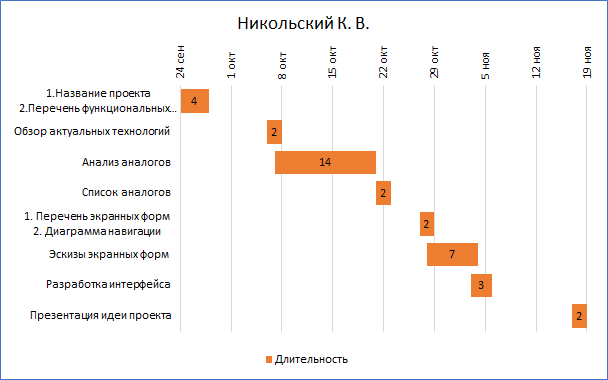


Рисунок 1 – Диаграмма Гантта для Никольского К. В.

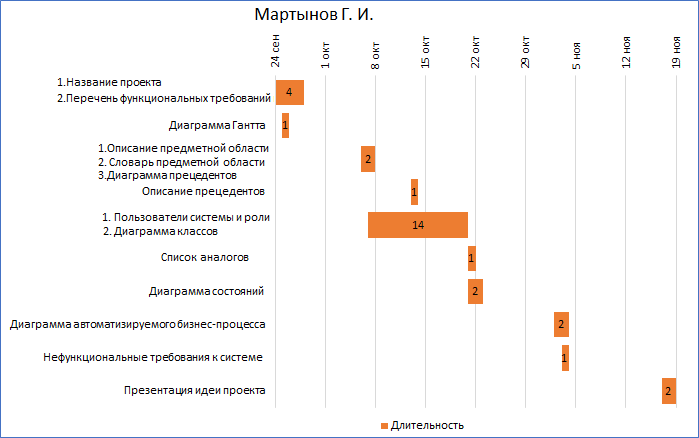
****

Рисунок 2 – Диаграмма Гантта для Мартынова Г. И.

**1.6 Обзор актуальных технологий**

1.6.1 Краткий обзор актуальных технологий для реализации компонентов системы

Для реализации данного проекта необходимо реализовать три модуля: модуль регистрации/авторизации, модуль хранения/предоставления данных, модуль представления. Рассмотрим, возможные технологии для реализации каждого модуля.

Модуль регистрации/авторизации (рабочее название - Insignia), является web-api сервисом, который имеет доступ к БД, с данными необходимыми для процедуры регистрации/авторизации. Основной сценарий работы данного компонента заключается в создании JWT, и последующей отправки его пользователю, или валидации JWT.

Модуль хранения/предоставления данных (рабочее название - StuffKeeper), является web-api сервисом, который имеет доступ к БД, которая хранит все данные бизнес-логики приложения. Основной сценарий работы заключается в обработке запроса на получение данных, проверке возможности доступа исходя из роли клиента (в случае необходимости обращается к Insignia), возвращение результата. Поскольку данный компонент также является web-api сервисом, стек технологий аналогичен предыдущему компоненту.

Модуль представления (рабочее название - Canvas) - это мобильное приложение под платформы IOS и Android. Основной сценарий работы данного компонента, это обращение к различным сервисам, парсинг полученных данных, и их отображение. Возможные технологии: Native (высокое быстродействие, разная кодовая база для платформ), Ionic (низкое быстродействие, общая кодовая база для платформ), React Native (высокое быстродействие, общая кодовая база для платформ, хорошая поддержка сообщества).

1.6.2 Выводы о применимости рассматриваемых технологий

Компоненты Insignia и StuffKeeper, являются веб-сервисами, из этого следует что технолгии для них будут одинаковыми. Реализовать их можно при помощи C#(ASP.NET Core), Java(Spring) или JS (Node.js + Express). Первые два подхода, дают надежное, быстрое, легко маштабируемое решение. Подход с использованием JS, позволяет реализовать сервис в короткие сроки, но такой проект проигрывает в быстродействии описанным ранее решениям. Выбирая между Java и C#, было отдано предпочтение C#, по причине более детального знакомства с платформой ASP.NET, и качественной поддержки продукта компанией Microsoft.

Компонент Canvas было решено разрабатывать с использованием React Native, из-за возможности использования общего кода для двух платформ без явного проигрыша в быстродействии.

**1.7 Анализ аналогов**

1.7.1 Перечень критериев оценки

1.7.2 Сравнительная таблица аналогов с балльной оценкой

1.7.3 Выводы по результатам анализа аналогов

2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

2.1 Описание предметной области

2.1.1 Данная система предназначена для создания собственного социального паспорта пользователя и автоматизации таких функций как создание и поиск: событий, заданий и аренды жилья. Имеются два вида пользователей: администратор и пользователь. Пользователь может создавать и пользоваться поиском: событий, заданий и жилья. Также ставить оценки в рейтинг и писать отзывы другим пользователям, редактировать свой социальный паспорт. Администратор имеет право удалять и замораживать пользователей и отменять события, задания и аренду жилья.

**2.2 Словарь предметной области**

2.2.1 Сущность «Пользователь (User)»

2.2.1.1 Авторизованный пользователь системы, который имеет логин, пароль, личные данные, ему доступны пользовательские функции системы.

2.2.1.2 Атрибуты:

− фамилия, имя, отчество;

− социальный паспорт;

− дата рождения;

− телефон;

− логин;

− пароль;

− адрес электронной почты;

− привилегии (уровень доступа: минимальный, средний, полный).

2.2.2 Сущность «Событие (Event)»

2.2.2.1 Пользовательская функция системы, которую можно создать, просматривать, редактировать, и удалять.

2.2.2.2 Атрибуты:

− стоимость;

− дата события.

2.2.3 Сущность «Задание (Task)»

2.2.3.1 Пользовательская функция системы.

2.2.3.2 Атрибуты:

− вознаграждение (обязательный параметр);

− категория (тяжелое/легкое, быстрое/долгое, срочное/не срочное).

2.2.4 Сущность «Жилье (Housing)»

2.2.4.1 Пользовательская функция системы.

2.2.4.2 Атрибуты:

− стоимость;

− кол-во свободных мест.

2.2.5 Сущность «Социальный паспорт (Social Passport)»

2.2.2.1 Пользовательская функция системы.

2.2.5.2 Атрибуты:

− рейтинг пользователя (0-1000 б., от новичка до просветленного. Считается исходя из следующих параметров:

• языки (перечислить владение языками);

• рейтинг знания языка (0-100 б.);

• красноречие (0-200 б., от новичка до просветленного);

• начитанность (0-200 б.);

• ответственность (0-100 б.);

• вежливость (0-100 б.);

• доброта (0-100 б.);

• уровень интеллекта (0-150 б.);

• текущий семейный статус (0-50 б.).).

2.2.7 Сущность «Администратор (Administrator)»

2.2.7.1 Пользователь системы, наследует свое поведение от сущности пользователь.

2.2.7.2 Атрибуты:

− дата приема на работу;

− оклад.

2.3 Пользователи разрабатываемой системы и роли

2.3.1 Характеристика аудитории программной системы

Аудитория программной системы, являются люди, желающие организовывать и участвовать в мероприятиях, предлагающие и заинтересованные в заданиях или жилье.

2.3.2. Расчетное количество пользователей системы

Пользователей данной системы может быть неограниченное количество, одновременно взаимодействующих друг с другом и пользующихся функционалом системы.

2.3.3. Перечень ролей пользователей

Роль 1: Гость – не авторизованный пользователь, который может либо зарегистрироваться, либо если он зарегистрирован – авторизоваться.

Роль 2: Пользователь – авторизованный пользователь системы, имеющий доступ к пользовательским функциям системы.

Роль 3: Администратор – авторизованный пользователь системы, имеющий доступ ко всем функциям системы.

**2.4 Диаграмма прецедентов**

2.4.1 Диаграмма прецедентов представлена на рисунке 3.



Рисунок 3 – Диаграмма прецедентов

**2.4.2 Описание прецедентов**

2.4.2.1 Название прецедента «Создать»

Предусловие: пользователь выбирает в боковом меню пункт «Создать».

Действующее лицо: пользователь

Основной поток: открывается окно меню выбора, что можно создать: событие, задание, аренда жилья. Пользователь выбирает нужную ему функцию и переходит в следующую экранную форму для создания выбранной им функции.

2.4.2.2 Название прецедента «Событие»

Предусловие: пользователь выбрал в боковом меню пункт «Создать», выбрал функцию «Событие».

Действующее лицо: пользователь

Основной поток: открывается о экранная форма для создания события. Пользователь заполняет все обязательные поля и нажимает на кнопку «Сохранить». Данные сохраняются в БД.

2.4.2.3 Название прецедента «Редактировать»

Предусловие: пользователь создал какую-то из функций. Он выбирает пункт «Редактировать» в меню созданной им функции.

Действующее лицо: пользователь

Основной поток: открывается окно редактирования созданной им функции.  
Он вносит нужные изменения, нажимает на кнопку «Сохранить». Данные сохраняются в БД.

2.4.2.4 Название прецедента «Посмотреть найденное»

Предусловие: пользователь хочет посмотреть имеющиеся события/задания/жилье, созданные другими пользователями. Он воспользовался пунктом бокового меню «Найти», выбрал нужную функцию. Нажал на заинтересовавший его пост.

Действующее лицо: пользователь

Основной поток: открывается форма просмотра выбранного им поста. Пользователь может откликнуться, добавить в понравившиеся, поставить рейтинг этому событию и написать отзыв о нем.

2.4.2.5 Название прецедента «Посмотреть список пользователей»

Предусловие: выбирает в боковом меню пункт «Посмотреть список пользователей».

Действующее лицо: администратор

Основной поток: открывается форма просмотра списка пользователей, их рейтинги, отзывы по их постам. Он может заморозить на какое-то время или удалить пользователя

2.5 Описание документов/сущностей предметной области

2.5.1 Диаграмма классов

2.5.1.1 Диаграмма классов представлена на рисунке 4.

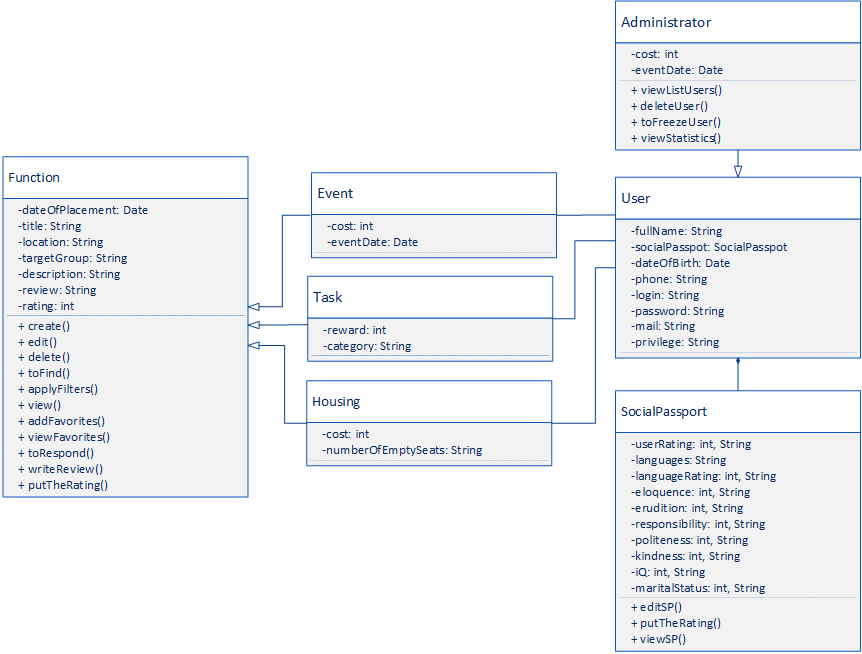


Рисунок 4 – Диаграмма классов

**2.5.2 Диаграмма состояний**

2.5.2.1 Описание бизнес-процесса работы ИС создание события представлено ниже:

1. Пользователь №1 создал событие, указав название, дату проведения, локацию, целевую группу, стоимость и описание события. Данные, указанные пользователем при оформлении события, передается во внешнюю систему проверки; событие ожидает подтверждения о проверке.

2. От системы проверки пришло сообщение о неверном вводе нескольких полей. У пользователя на экране появляется соответствующее уведомление, он обязан исправить эти поля.

3. От системы проверки пришло подтверждение корректности ввода данных. Система размещает данное событие в список событий, статус события ''Размещено'.

4. Событие не привлекло ни одного пользователя к моменту проведения, оно закрывается и его статус изменяется на «Отменено».

5. Пользователь №2 находит событие, и жмет «Откликнуться», статус события меняется на «Откликнулся 1 человек».

6. К моменту проведения, события статус меняется на «Идет сейчас»

7. По завершению события, статут меняется на «Завершено»

3.5.2 Диаграмма состояний, полученная на основе диаграммы классов для объекта «Заявка» представлена на рисунке 5.

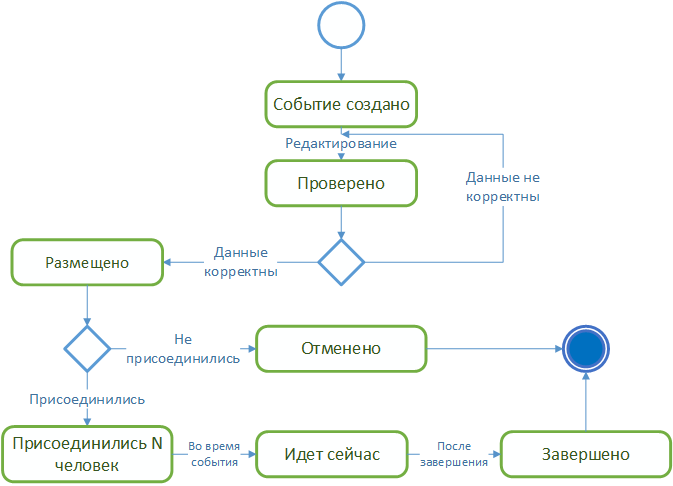


Рисунок 5 – Диаграмма состояний для объекта «Событие»

2.6 Анализ бизнес-процессов

2.6.1 Диаграмма автоматизируемого бизнес-процесса «Посмотреть события» представлена на рисунке 6.

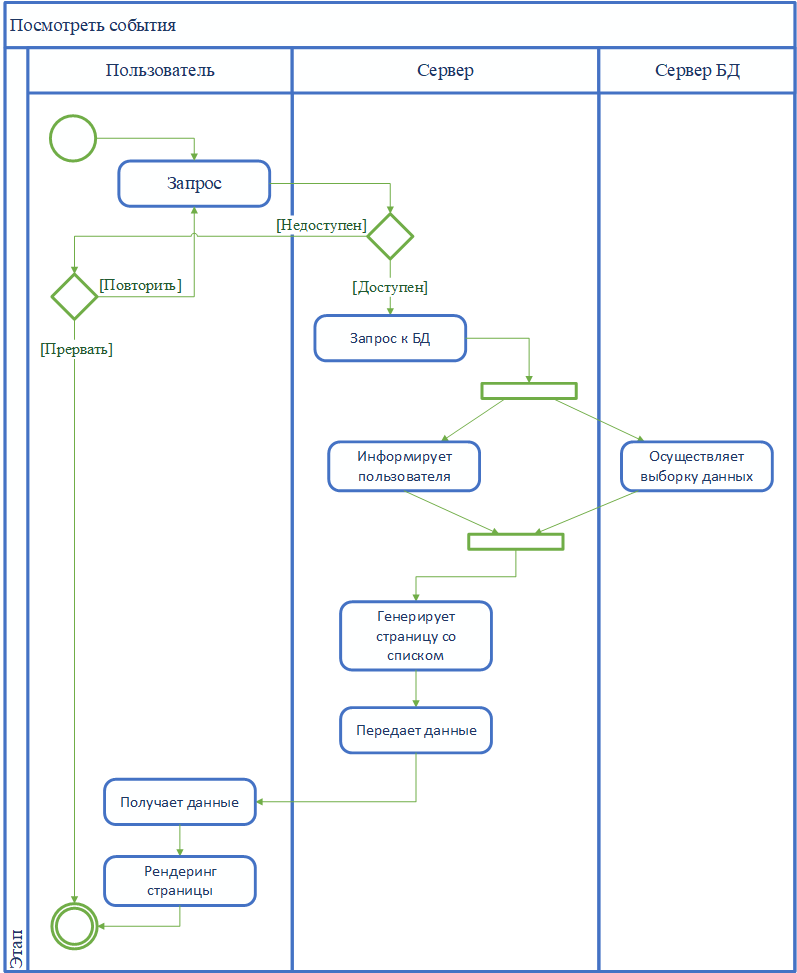


Рисунок 5 – Диаграмма автоматизируемого бизнес-процесса

2.6.2 Описание зон ответственности пользователей системы (ролей)

2.8 Функциональные требования к системе

2.8.1 Гость:

− Зарегистрироваться;

− Войти в систему.

2.8.2 Пользователь:

− Редактировать свой соц. паспорт;

− Создать ⇒ (Редактировать, Отменить) / Найти ⇒ (Применить фильтры, Присоединиться, Написать отзыв, Поставить рейтинг):

1. Событие;
2. Задание;
3. Жилье.

2.8.3 Администратор наследует пользователя, а также может:

− Удалить пользователя;

− Посмотреть статистики.

2.9 Нефункциональные требования к системе

2.9.1 Перечень нефункциональных требований к системе

2.9.1.1

3 МАКЕТ ПРОГРАММНОЙ СИСТЕМЫ

3.1 Описание принципов организации интерфейса

3.2 Структура интерфейса, навигация

3.2.1 Перечень экранных форм

3.2.2 Диаграмма навигации

3.2 Эскизы интерфейса пользователя

Эскизы экранных форм (mockup)

4 РАЗРАБОТКА ПРОТОТИПА ПРОГРАММНОЙ СИСТЕМЫ