**САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**ПЕТРА ВЕЛИКОГО**

**Институт компьютерных наук и технологий**

**Кафедра «Распределенные вычисления и компьютерные сети»**

**КУРСОВОЙ ПРОЕКТ**

**Мобильное приложение для проверки погоды**

по дисциплине «Проектирование программных систем»

Выполнил

студент гр. <*подпись*> В.Б.Борисов

43507/1

Руководитель

<*подпись*> К.В.Соколов

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2016 г.

Санкт-Петербург

2016

**User stories**

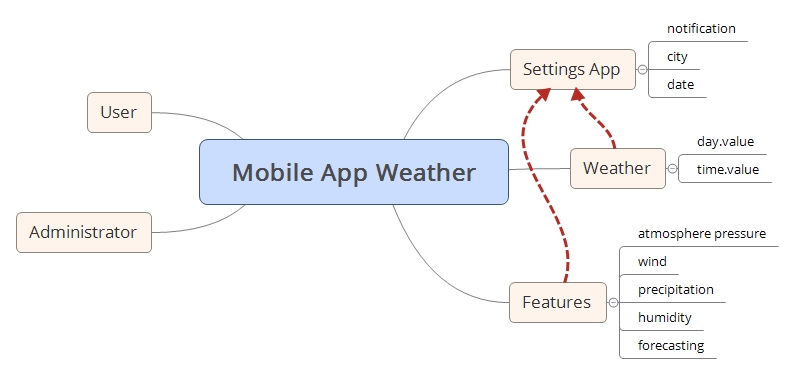
As an Application user, I am capable of:

1. Checking the weather;
2. Changing the city;
3. Changing date;
4. Changing widgets;
5. Weather forecasting(today, tomorrow, whole week);
6. Checking atmosphere pressure;
7. Checking wind;
8. Checking precipitation (rain,show);
9. Checking humidity;
10. Setting notification (for example, sudden change weather);
11. Sending data to server, if weather forecast is wrong;

As an application administrator, I am capable of:

1. Setting new data for every city;
2. Setting new weather features;
3. Approve updates;
4. Changing existing data;
5. Changing existing weather features;
6. Getting data from user (location);
7. View the details of the actions on the device

**Сущности предметной области**

****

User – пользователь

Administrator - администратор

Mobile App Weather – приложение предоставляющее пользователю интерфейсы для доступа к данным погоды (Weather), особенностям погоды (Weather features), настройкам пользователя (Application settings).

Features – особенности погоды, зависит от настроек пользователя (город и дата)

Weather – основной экран приложения отображающий дневную погоду и часовую погоду, зависит от настроек пользователя (город и дата)

Settings App – настройки пользователя: город, дата, уведомления. Не имеет зависимостей.

**Компоненты системы**

App Settings:

private

<string> city

<datetime>date

<mytime>notification

public functions

<string> city

set city

get city

<datetime> date

set date

get date

<mytime> notification

set notification time

set notification frequency

set notification off

<Setting App>getInstance

Weather

private:

<interval> day.value

vector<interval>hour.value

<Settings App> set

Public function:

<interval>day.value

get day.value

set day.value (only for admin)

vector<interval>hour.value

get hour.value

set day.value (only for admin)

<Settings App>setInstance

<data>getdatafromserver(server address,city,date)

Features

private

<float>wind

<float>atmosphere pressure

<float>precipation

<float> humidity

<Settings App>set

public functions

<float>wind

Get wind

Set wind (only for admin)

<float>atmosphere pressure

Get atmosphere pressure

Set atmosphere pressure(only for admin)

<float>precipation

Get precipation

Set precipation (only for admin)

<float>humidity

Get humidity

Set humidity(only for admin)

<Settings App>setInstance

<data>getdatafromserver(server address,city,date)

**Реализация**

Для решение данной задачи структура client-server отлично подходит. На самом сервере будут проводиться различные вычисления, также на нем будет храниться основная информация по всем городам и ближайшим датам. На сервере проще организовать контроль полномочий, чтобы разрешать доступ к данным только клиентам с соответствующими правами доступа, а администратор будет поддерживать работоспособность самого сервера. После запуска, приложение будет автоматически запрашивать данные с сервера n раз за m времени.

Управление зависимостями: В проекте реализована зависимость классов weather и features от класса settings app, но они получают неизменяемый (для этих классов) инстанс. Следовательно, в данной реализации классы являются самодостаточными, и изменение одного класса по сути не влияет на остальные классы.

Тестируемость: Классы weather и weather features можно протестировать, подав им mock object application settings и перегрузить функцию getDataFromServer на функцию возвращающую корректную структуру данных. Класс settings app самодостаточный и может быть протестирован вне общего проекта.

**Используемые шаблоны**

Начнем с класса Settings App. Для данного класса подойдет шаблон Singleton. Для того чтобы мы не смогли получить данные из класса weather отличающиеся по дате или городу от данных из класса features.

Если мы захотим добавить графическое оформление для приложения, то будет удобно использовать шаблон Flyweight, чтобы сохранить готовые наборы объектов для разных условий погоды.

Для изменения состояния классов weather и features мы можем воспользоваться шаблоном Observer, чтобы сообщать об изменении города или даты.

Для отключения функций оповещения о погоде можно воспользоваться паттерном state.