МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине «Разработка программного обеспечения ИС»

Тема: Анализ данных Jira – статистика по доскам

Студенты гр. 6303 _____ Куценок Л.А.

_____ Сергеенков М.Ю.

_____ Гречков С.Д.

_____ Фокин К.С.

Преподаватель _____ Заславский М.М.

Санкт-Петербург

2019

ЗАДАНИЕ

Студенты Куценок Л.А., Сергеенков М.Ю., Гречков С.Д., Фокин К.С.

Группа 6303

Тема проекта: Разработка приложения для анализа данных из Atlassian Jira и построения статистики по доскам

Исходные данные:
Необходимо реализовать приложение, использующее СУБД MongoDB

Содержание пояснительной записки:
«Содержание», «Введение», «Качественные требования к решению», «Сценарий использования», «Модель данных», «Разработанное приложение», «Заключение», «Список использованных источников»

Предполагаемый объем пояснительной записки:
Не менее 10 страниц.

Дата выдачи задания:

Дата сдачи реферата:

Дата защиты реферата:

Студенты гр. 6303	_____	Куценок Л.А.
	_____	Сергеенков М.Ю.
	_____	Гречков С.Д.
	_____	Фокин К.С.
Преподаватель	_____	Заславский М.М.

АННОТАЦИЯ

В ходе выполнения проекта было необходимо разработать приложение для анализа данных из Atlassian Jira и построения статистики по доскам, позволяющее пользователю просматривать аналитику оценки трудозатрат по проектам и исполнителям, фактических трудозатрат по направлению и тенденции трудозатрат по заказчикам за различные временные периоды, категории и проекты, а также реализовать возможность импорта/экспорта данных приложения и загрузка сведений из Atlassian Jira.

SUMMARY

The main objective of the project was to develop an application for analyzing data from Atlassian Jira and building statistics of the boards, allowing the user to view the analytics of estimatated labor costs for projects and employees, actual labor costs for different issue types and labor trends for customers for different time periods, categories and projects, and also to implement the ability to import / export application data and receive row data from Atlassian Jira.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель работы – создать сервис, позволяющий подробно проанализировать оценить качество работы команды на основе данных из Atlassian Jira.

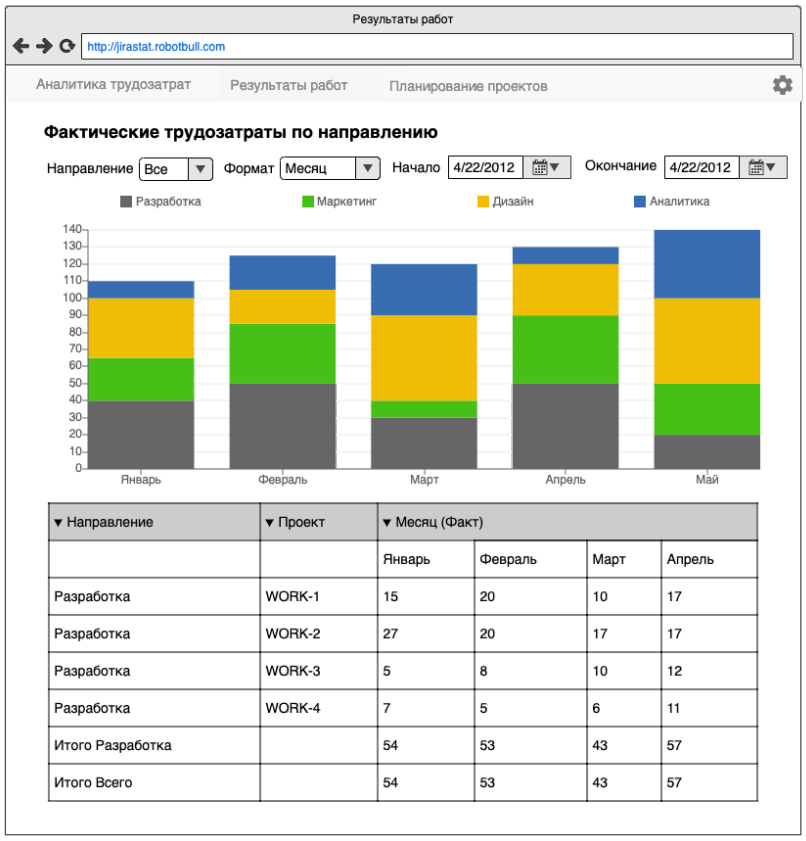
Для реализации поставленной задачи было разработано веб-приложение, которое позволяет отображать данные и графики по различным показателям, при этом позволяющее удобно с ними взаимодействовать.

1. КАЧЕСТВЕННЫЕ ТРЕБОВАНИЯ К РЕШЕНИЮ

Требуется реализовать приложение, использующее СУБД MongoDB.

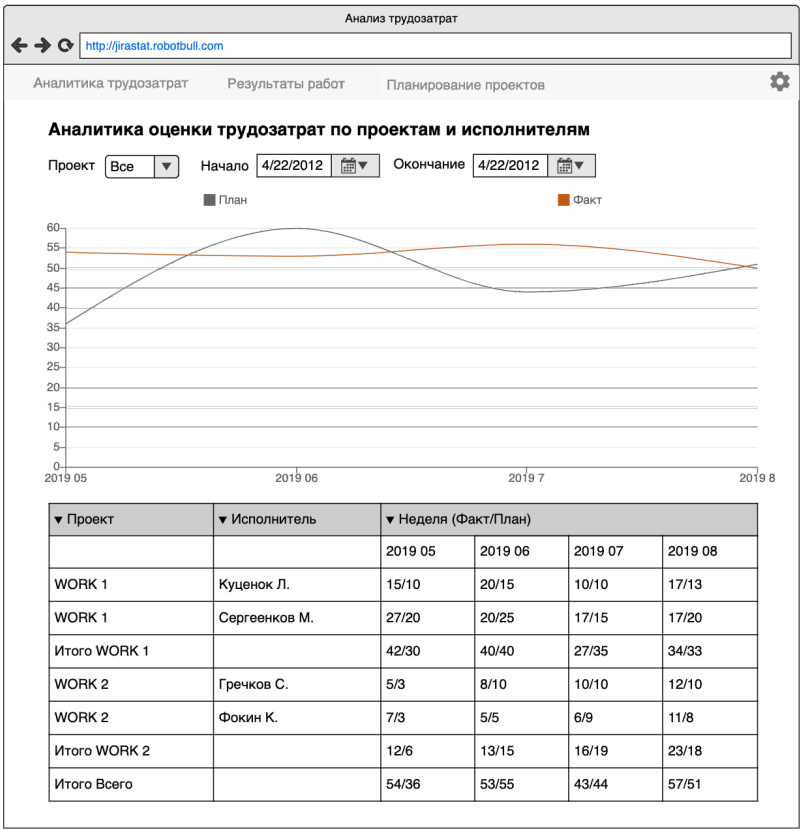
2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

2.1. Макеты пользовательского интерфейса



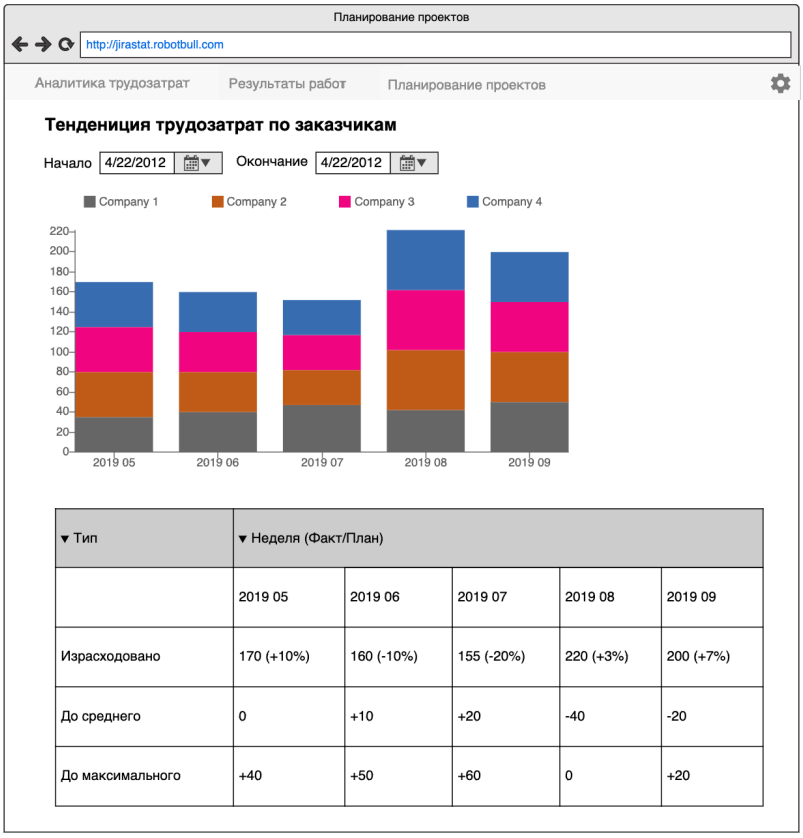
▾ Направление	▾ Проект	▾ Месяц (Факт)			
		Январь	Февраль	Март	Апрель
Разработка	WORK-1	15	20	10	17
Разработка	WORK-2	27	20	17	17
Разработка	WORK-3	5	8	10	12
Разработка	WORK-4	7	5	6	11
Итого Разработка		54	53	43	57
Итого Всего		54	53	43	57

Рис. 1. Экран просмотра фактических трудозатрат по направлению



▾ Проект	▾ Исполнитель	▾ Неделя (Факт/План)			
		2019 05	2019 06	2019 07	2019 08
WORK 1	Куценок Л.	15/10	20/15	10/10	17/13
WORK 1	Сергеенков М.	27/20	20/25	17/15	17/20
Итого WORK 1		42/30	40/40	27/35	34/33
WORK 2	Гречков С.	5/3	8/10	10/10	12/10
WORK 2	Фокин К.	7/3	5/5	6/9	11/8
Итого WORK 2		12/6	13/15	16/19	23/18
Итого Всего		54/36	53/55	43/44	57/51

Рис. 2. Экран просмотра аналитики оценки трудозатрат по проектам и исполнителям



▾ Тип	▾ Неделя (Факт/План)				
	2019 05	2019 06	2019 07	2019 08	2019 09
Израсходовано	170 (+10%)	160 (-10%)	155 (-20%)	220 (+3%)	200 (+7%)
До среднего	0	+10	+20	-40	-20
До максимального	+40	+50	+60	0	+20

Рис. 3. Экран просмотра тенденции трудозатрат по заказчикам

2.2. Описание сценариев использования

2.2.1. Сценарий использования "Оценить эффективность работы сотрудников и корректность планируемых трудозатрат"

Основной сценарий:

- Пользователь зашёл на сайт
- Пользователь нажимает на кнопку "Аналитика трудозатрат" в верхнем меню
- Происходит перенаправление на страницу "Аналитика оценки трудозатрат по проектам и исполнителям"
- Пользователю выводится график планируемых и фактических трудозатрат по всем проектам за последний 4 недели, а также связанная с графиком таблица данных, в которой указаны фактические и планируемые трудозатраты по каждому исполнителю внутри каждого проекта

Альтернативный сценарий

- Пользователь изменяет нужный проект и/или нужный период выборки
- График и таблица данных обновляются в соответствии с выбранными параметрами

2.2.2. Сценарий использования "Оценить фактические трудозатраты по направлению"

Основной сценарий:

- Пользователь зашёл на сайт
- Пользователь нажимает на кнопку "Результаты работ" в верхнем меню
- Происходит перенаправление на страницу "Фактические трудозатраты по направлению"

- Пользователю выводится график фактических трудозатрат по всем направлениям работы компании в сумме за последние 3 месяца, а также связанная с графиком таблица данных.

Альтернативный сценарий:

- Пользователь изменяет нужное направление, формат группировки (неделя/месяц) и/или нужный период выборки
- График и таблица данных обновляются в соответствии с выбранными параметрами

2.2.3. Сценарий использования "Оценить тенденцию трудозатрат по проектам"

Основной сценарий:

- Пользователь зашёл на сайт
- Пользователь нажимает на кнопку "Планирование проектов" в верхнем меню
- Происходит перенаправление на страницу "Тенденция трудозатрат по проектам"
- Пользователю выводится график фактических трудозатрат за последние 4 недели по проектам в сумме для прошедших недель и планируемых трудозатрат по проектам в сумме для будущих недель, а также связанная с графиком таблица данных, в которой указаны и планируемые фактические трудозатраты по каждому направлению отдельно за каждую неделю, с отражением отклонения от среднего и максимального объема работ

Альтернативный сценарий:

- Пользователь изменяет нужный период выборки
- График и таблица данных обновляются в соответствии с выбранными параметрами

3. МОДЕЛЬ ДАННЫХ

3.1. NoSQL модель данных

3.1.1. Подробное описание и расчёт объема

В качестве СУБД используется MongoDB[1], в которой содержится единственная коллекция issue, отвечающая за хранение данных о задачах из Atlassian Jira.

Каждый документ содержит следующие поля:

- **_id** – уникальный идентификатор документа. Тип - ObjectID. V = 12b
- **key** – идентификатор задачи в Jira. Тип - String. V = 2b * N, где N - средняя длина ключа задачи. На основе загруженных данных установлено, что N = 9. Тогда V = 18b
- **created** – дата и время создания задачи в Jira. Тип - date. V = 8b
- **duedate** – срок выполнения задачи в Jira. Тип - date. V = 8b
- **resolutiondate** – дата закрытия задачи в Jira. Тип - date. V = 8b
- **assignee** – установленный в Jira исполнитель задачи. Тип - string. V = 2b * N, где N - средняя длина имени исполнителя. На основе текущих загруженных установлено, что N = 10. Тогда V = 20b
- **timespent** – время, затраченное на выполнение задачи (в секундах). Тип - int. V = 4b
- **timeoriginalestimate** – первоначальное предполагаемое время выполнения задачи (в секундах). Тип - int. V = 4b
- **status** – установленный в Jira статус задачи. Тип - string. V = 2b * N, где N - средняя длина названия статуса. На основе загруженных данных установлено, что N = 8. Тогда V = 16b
- **component** – установленная в Jira компонента, которой принадлежит задачи. Тип - string. V = 2b * N, где N - средняя длина названия

компоненты. На основе загруженных данных установлено, что N = 11. Тогда V = 22b

- **project** – установленный в Jira эпик задачи. Тип - string. V = 2b * N, где N - средняя длина ключа эпика. На основе загруженных данных установлено, что N = 7. Тогда V = 14b
- **category** – установленный в Jira тип задачи. Тип - string. V = 2b * N, где N - средняя длина названия типа. На основе загруженных данных установлено, что N = 8. Тогда V = 16b

Итого объем документа в среднем должен составлять 150b. Объем всей БД V = 150b * N, где N количество записей о задачах в БД.

На основе существующих данных рассчитано, что в среднем за один месяц создается 37 задач, таким образом средний объем БД за один месяц составляет 5550b = 5.42kb.

Аналогично, средний объем БД за один год составляет 65kb.

3.1.2. Примеры запросов

- Запрос на добавление нового документа в коллекцию:

```
db.issue.insert_one({
        'key': 'DEV-1',
        'created': '2019-01-01 10:00:00',
        'duedate': '2019-01-10 14:00:00',
        'resolutiondate': '2019-01-08 15:30:00',
        'assignee': 'i.ivanov',
        'timespent': 7200,
        'timeoriginalestimate': 3600,
        'status': 'Готово',
        'component': 'Company',
        'project': 'PROJECT-1',
        'category': 'Разработка'
    })
```

Пример исходного кода запроса к MongoDB на добавление нового документа в коллекцию

Для добавления нового документа в базу требуется только один запрос. Таким образом, для обработки N задач из Jira требуется N запросов.

- Запрос на поиск всех задач, закрепленных за исполнителем:

```
db.issue.find({'assignee': 'i.ivanov'})
```

Пример исходного кода запроса к MongoDB на поиск всех задач, закрепленных за исполнителем

- Запрос на получение суммы затраченных на задачи часов, сгруппированных по месяцу и категории задачи:

```
db.getCollection('issue').aggregate(
  [
      {
          $group: {
              _id: {
                  category: "$category",
                  month: {
                      $dateToString: {
                          date: "$created",
                          format: "%m %Y"
                      }
                  }
              },
              total: {
                  $sum: "$timespent"
              }
          }
      },
      {
          $group: {
              _id: {
                  category: "$_id.category"
              },
              hours: {
                  $push: {
                      k: "$_id.month",
                      v: "$total"
                  }
              }
          }
      },
      {
          $addFields: {
              hours: {
                  $arrayToObject: "$hours"
              }
          }
      },
      {
          $project: {
              workType: "$_id.category",
              hours: 1,
              _id: 0
          }
      }
   ]
)
```

Пример исходного кода запроса к MongoDB на получение суммы затраченных на задачи часов, сгруппированных по месяцу и категории задачи

Результат:


```
{
  "hours": {
      "04 2019": 39600,
      "03 2019": 127800,
      "02 2019": 72000,
      "05 2019": 18000
  },
  "workType": "Дизайн"
},
{
  "hours": {
      "02 2019": 0,
      "03 2019": 0
  },
  "workType": "Администрирование"
}
{
  "hours": {
      "01 2019": 109800,
      "11 2018": 75600,
      "12 2018": 73800,
      "10 2018": 124200,
      "09 2018": 45000,
      "03 2019": 236340,
      "04 2019": 184500,
      "05 2019": 137700,
      "02 2019": 172800
  },
  "workType": "Разработка"
}
```


Пример результата запроса к MongoDB на получение суммы затраченных на задачи часов, сгруппированных по месяцу и категории задачи

Время выполнения:

- N = 1000, t = 6ms
- N = 2000, t = 9ms
- N = 3000, t = 12ms
- N = 4000, t = 17ms
- N = 5000, t = 19ms
- N = 6000, t = 25ms

Из чего можно сделать вывод, что время выполнения линейно и на каждую 1000 записей необходимо ~4ms

3.2. SQL модель данных

3.2.1. Подробное описание и расчёт объема

В качестве СУБД используется для демонстрации SQL модели используются MySQL[2], в которой содержится база данных jira-stats и единственная таблица issue, отвечающая за хранение данных о задачах из Atlassian Jira.

Структура таблицы issue повторяет аналогичную коллекцию из MongoDB и содержит следующие поля:

- **id**. Тип - Int. V = 4b
- **key**. Тип - Varchar. V = 2b * N, V = 18b
- **created**. Тип - Datetime. V = 8b
- **duedate**. Тип - Datetime. V = 8b
- **resolutiondate**. Тип - Datetime. V = 8b
- **assignee**. Тип - String. V = 2b * N, V = 20b
- **timespent**. Тип - Int. V = 4b
- **timeoriginalestimate**. Тип - Int. V = 4b
- **status**. Тип - String. V = 2b * N, V = 16b
- **component**. Тип - string. V = 2b * N, V = 22b
- **project**. Тип - String. V = 2b * N, V = 14b
- **category**. Тип - String. V = 2b * N, V = 16b

Итого объем документа в среднем должен составлять 142b. Объем всей БД V = 142b * N, где N количество записей о задачах в БД. За месяц: 5254b = 5.13kb. За год: 61kb.

3.2.2. Примеры запросов

- Запрос на добавление нового документа в коллекцию:

```
INSERT INTO `issue` (
   key,
```

```
    created,
    duedate,
    resolutiondate,
    assignee,
    timespent,
    timeoriginalestimate,
    status,
    component,
    project,
    category
) VALUES (
    'DEV 1',
    '2019-01-01 10:00:00',
    '2019-01-10 14:00:00',
    '2019-01-08 15:30:00',
    'ivanov.i',
    7200,
    3600,
    'Готово',
    'Company',
    'PROJECT-1',
    'Разработка'
);
```

Пример исходного кода SQL-запроса на добавление новой записи в таблицу

Для добавления нового документа в базу требуется только один запрос. Таким образом, для обработки N задач из Jira требуется N запросов.

- Запрос на поиск всех задач, закрепленных за исполнителем:

```
SELECT * FROM issue WHERE assignee = 'i.ivanov'
```

Пример исходного кода SQL-запроса на на поиск всех задач, закрепленных за исполнителем

- Запрос на получение суммы затраченных на задачи часов, сгруппированных по месяцу и категории задачи:

```
SELECT
    DATE_FORMAT( resolutiondate, "%m %Y" ) AS MONTH,
    category,
    sum( timespent ) AS timespent
FROM
    issue
GROUP BY
    MONTH ( resolutiondate ),
    Category
```

Пример исходного кода SQL-запроса на получение суммы затраченных на задачи часов, сгруппированных по месяцу и категории задачи

Результат:

```
01 2019      Разработка           111600
02 2019      Администрирование    0
02 2019      Дизайн               72000
02 2019      Разработка           122400
03 2019      Администрирование    0
03 2019      Дизайн               112680
03 2019      Разработка           248940
04 2019      Дизайн               96480
04 2019      Разработка           242100
05 2019      Дизайн               25200
05 2019      Разработка           162900
09 2018      Разработка           37800
10 2018      Разработка           108000
11 2018      Разработка           43200
12 2018      Разработка           82800
```

Пример результата SQL-запроса на получение суммы затраченных на задачи часов, сгруппированных по месяцу и категории задачи

Время выполнения:

- N = 1000, t = 3ms
- N = 2000, t = 5ms
- N = 3000, t = 6ms
- N = 4000, t = 6ms
- N = 5000, t = 8ms
- N = 6000, t = 8ms

Из чего можно сделать вывод, что время выполнения линейно и на каждую 1000 записей необходимо ~1ms

3.3. Сравнение NoSQL и SQL моделей данных

- По объему данных: объем базы данных примерно равен для обеих моделей, так как структура сущностей проектируемой модели одинакова как в MongoDB, так и в MySQL, а также сама модель содержит всего одну коллекцию/таблицу
- По количеству запросов: количество запросов на вставку и выборку сущностей из БД одинаково по причинам, изложенным в предыдущем пункте

- По времени выполнения запросов: как видно по представленным выше результатам, выполнение запроса на получение суммы затраченных на задачи часов, сгруппированных по месяцу и категории задачи, в случае MySQL выполняется несколько быстрее, чем с помощью MongoDB. Однако необходимо обратить внимание, что запросы к MongoDB позволяют производить многоуровневую группировку на уровне БД, а в случае MySQL эту группировку придется производить на программном уровне, что в сумме сделает обработку данных из MySQL дольше, чем с помощью MondoDB. Кроме того, представленный запрос использует 2 уровня вложенности, в то время как более сложные запросы, используемые в проекте, используют до 4-х уровней вложенности. Таким образом, для избежания накладных расходов по обработке результатов запросов рациональнее использовать MongoDB в качестве СУБД

4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ

4.1. Краткое описание

Для упрощения процесса разработки было принято решение разделить приложение на frontend и backend части

Frontend-часть реализована с использованием JavaScript фреймворков React[3] и CoreUI[4]. Данные для построения графиков и таблиц, импорта и экспорта данных передаются с помощью REST API на Backend-сервер.

Backend-часть приложения реализована с использованием Python фреймворка Flask[5] и представляет собой набор API методов, позволяющих Frontend-приложению загружать данные из БД с помощью библиотеки PyMongo[6] для отображения пользователю, а также функционал, требуемый для получения исходных данных из Atlassian Jira.

4.2. Схема экранов приложения

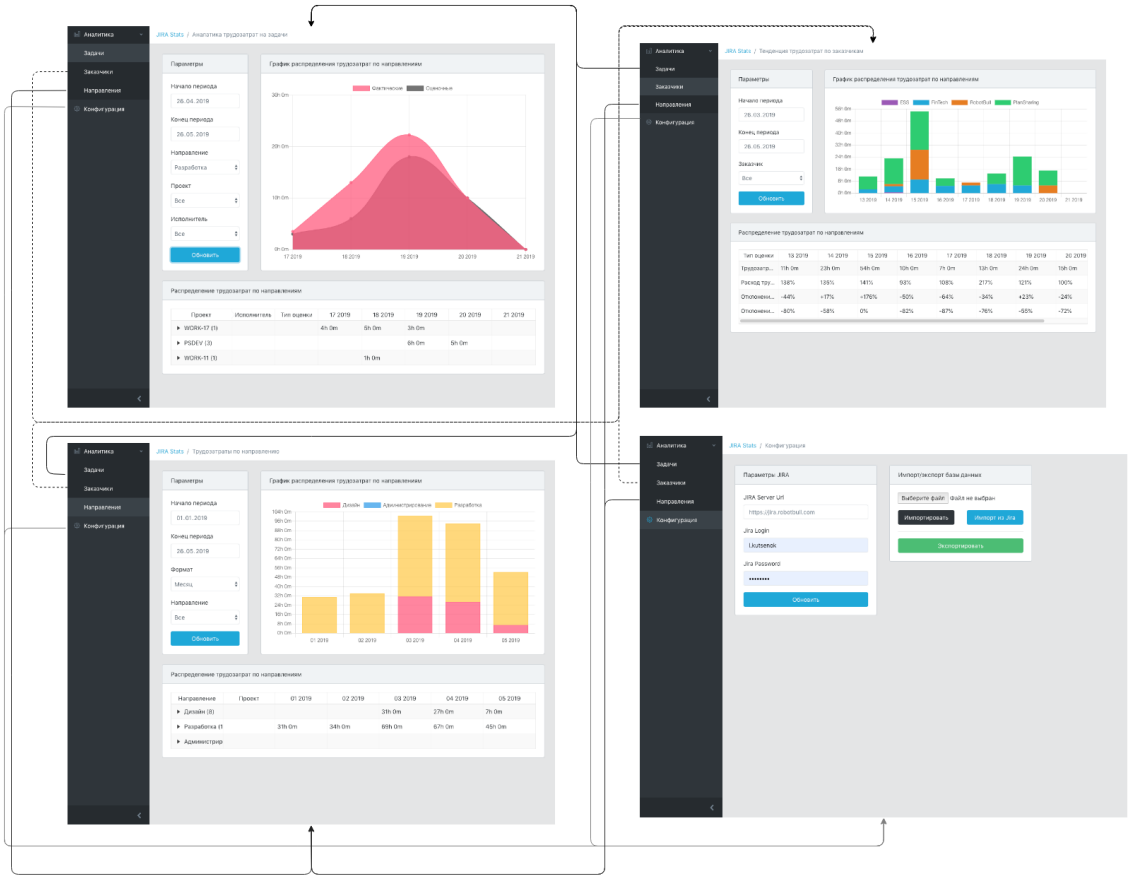


Рис. 4. Схема экранов приложения

4.3. Использованные технологии

БД: MongoDB

Backend: Python 3.6, PyMongo, python-jira

Frontend: React 16, CoreUI, Axios

4.4. Ссылка на приложение

Ссылка на приложение доступна в разделе “Список использованных источников” [7]

ЗАКЛЮЧЕНИЕ

В ходе выполнения проекта было разработано приложение для анализа данных из Atlassian Jira и построения статистики по доскам, позволяющее пользователю просматривать аналитику оценки трудозатрат по проектам и исполнителям, фактических трудозатрат по направлению и тенденции трудозатрат по заказчикам за различные временные периоды, категории и проекты, а также реализована возможность импорта/экспорта данных приложения и загрузка сведений из Atlassian Jira.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация MongoDB: https://docs.mongodb.com/manual/
2. Документация MySQL: https://dev.mysql.com/doc/
3. Документация React 16: https://reactjs.org/docs/getting-started.html
4. Документация CoreUI: https://coreui.io/docs/getting-started/introduction/
5. Документация Flask: http://flask.pocoo.org/docs/1.0/
6. Документация PyMongo: https://api.mongodb.com/python/current/
7. Исходный код приложения: https://github.com/moevm/nosql1h19-jira-stats