

**Министерство образования науки
Российской Федерации**

**Рязанский государственный радиотехнический университет
Кафедра ВПМ**

Учебная дисциплина: Мировые информационные ресурсы
(наименование дисциплины)

**Контрольная работа
Вариант 5
«Персональный сайт»**

Выполнил:

Студент группы: 4041

Бирюков Павел Юрьевич
(Ф.И.О.)

Принял:

Преподаватель: Коротаев

Александр Николаевич
(Ф.И.О.)

Оценка: _____

Рязань 2018

Оглавление

Задание	3
Используемые компоненты	3
Структура файлов.....	4
Листинг программы	6
db.json	6
planning-page.component.html.....	8
planning-page.component.ts	8
bill-card.component.html	10
bill-card.component.ts.....	11
currency-card.component.html	11
currency-card.component.ts	11
bill-page.component.html	12
bill-page.component.ts	12
bill.service.ts	13
base-api.ts.....	13
Результат работы	15

Задание

Написать персональный сайт.

Используемые компоненты

AngularJS. JavaScript-фреймворк с открытым исходным кодом. Предназначен для разработки одностраничных приложений. Его цель – расширение браузерных приложений на основе MVC-шаблона, а также упрощение тестирования и разработки.

Фреймворк работает с HTML, содержащим дополнительные пользовательские атрибуты, которые описываются директивами, и связывает ввод или вывод области страницы с моделью, представляющей собой обычные переменные JavaScript. Значения этих переменных задаются вручную или извлекаются из статических или динамических JSON-данных.

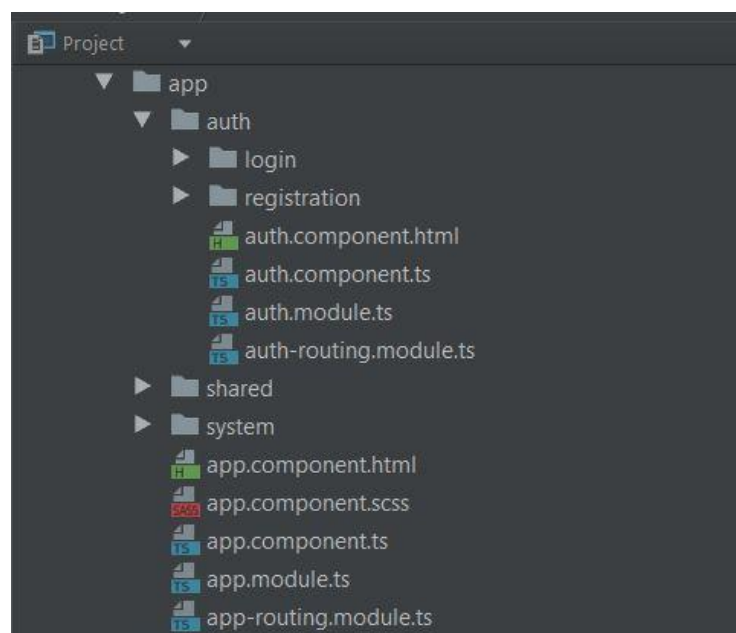
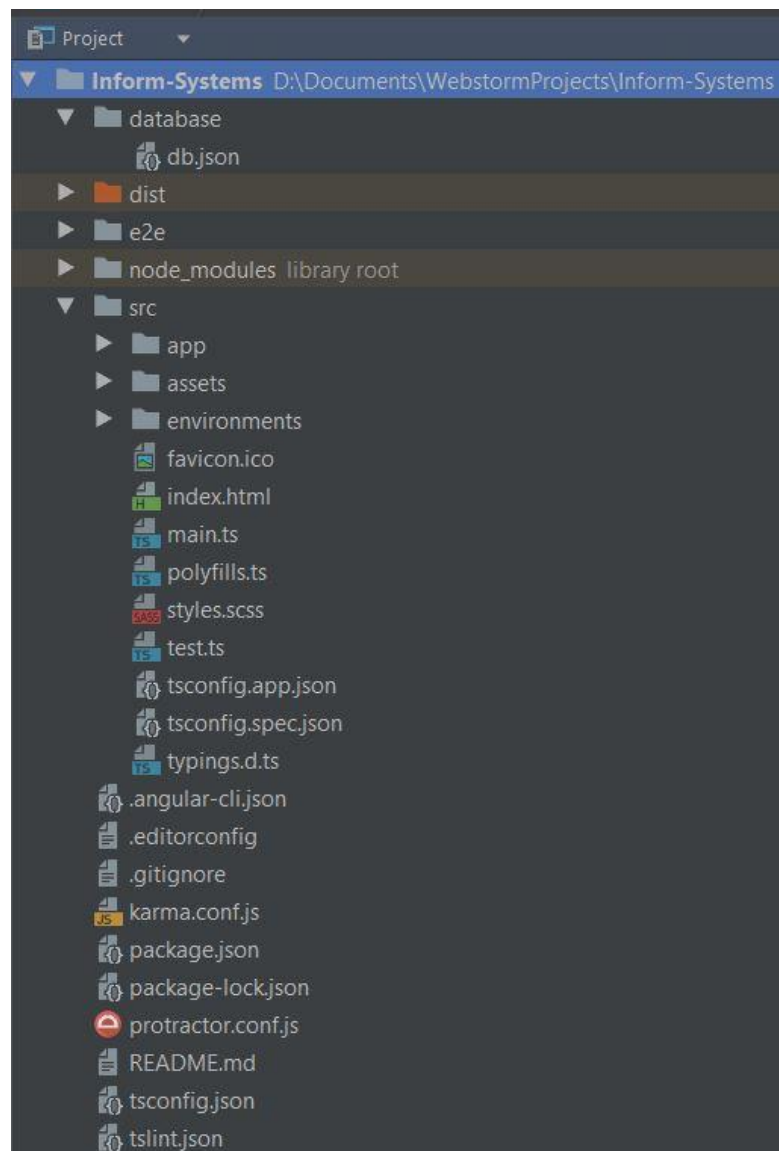
Bootstrap. Свободный набор инструментов для создания сайтов и веб-приложений. Включает в себя HTML и CSS-шаблоны оформления для типографики, веб-форм, кнопок, меток, блоков навигации и прочих компонентов веб-интерфейса, включая JavaScript-расширения.

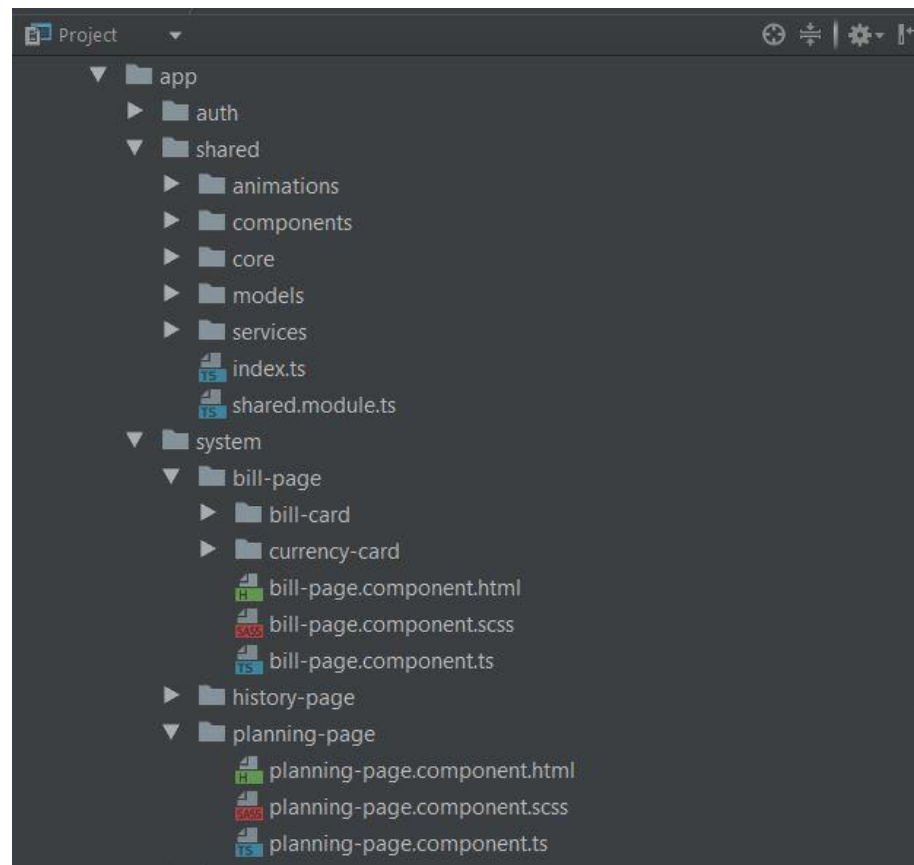
ngx-charts. Модуль, позволяющий строить диаграммы для Angular.

Moment.js. Это проект с открытым исходным кодом, позволяет упростить проверки, синтаксический анализ и обработку дат на стороне клиента. Moment.js избавляет от необходимости использовать родной JavaScript-объект Date напрямую. Библиотека является оболочкой для объекта Date, что позволяет значительно проще работать с объектом.

Font Awesome. Это SVG шрифт с иконками, которые могут добавляться к любым элементам веб-страницы, чтобы повысить их наглядность и улучшить дизайн.

Структура файлов





Листинг программы

Ниже будет приведен код двух модулей приложения:

- Страница планирования
- Страница счета. Объединяет в себе два компонента — **bill-card** и **currency-card**. Так же используется один сервис для получения курсов валют — **bill.service**.

db.json

```
{
  "users": [
    {
      "id": 1,
      "email": "admin@mail.ru",
      "password": "qwerty12",
      "name": "Павел"
    }
  ],
  "bill": {
    "value": 71300,
    "currency": "RUB"
  },
  "categories": [
    {
      "id": 1,
      "name": "Дом",
      "capacity": 15000
    },
    {
      "id": 2,
      "name": "Супермаркеты",
      "capacity": 11000
    },
    {
      "id": 3,
      "name": "Одежда и обувь",
      "capacity": 7000
    },
    {
      "id": 4,
      "name": "Развлечения",
      "capacity": 5000
    },
    {
      "id": 5,
      "name": "Перевод",
      "capacity": 15000
    },
    {
      "id": 6,
      "name": "Зарплата",
      "capacity": 0
    },
    {
      "id": 7,
      "name": "Остальное",

```

```

        "capacity": 3000
    },
    ],
    "events": [
        {
            "id": 1,
            "type": "income",
            "amount": 25000,
            "category": 6,
            "date": "15.01.2018 19:49:02",
            "description": "Аванс"
        },
        {
            "id": 2,
            "type": "outcome",
            "amount": 1300,
            "category": 1,
            "date": "08.01.2018 18:24:02",
            "description": "Покупка микроволновки"
        },
        {
            "type": "income",
            "amount": 1480,
            "category": 2,
            "date": "07.01.2018 15:00:18",
            "id": 3,
            "description": "Поход в магазин с баллами"
        },
        {
            "type": "outcome",
            "amount": 2470,
            "category": 2,
            "date": "07.01.2018 15:00:28",
            "id": 4,
            "description": "Закупка на неделю"
        },
        {
            "type": "outcome",
            "amount": 4000,
            "category": 3,
            "date": "02.01.2018 11:01:58",
            "id": 5,
            "description": "Покупка кроссовок"
        },
        {
            "type": "outcome",
            "amount": 300,
            "category": 4,
            "date": "08.01.2018 21:31:05",
            "description": "Кино",
            "id": 6
        },
        {
            "type": "outcome",
            "amount": 14200,
            "category": 5,
            "date": "08.01.2018 21:31:26",
            "description": "Ипотека",
            "id": 7
        },
        {
            "type": "outcome",
            "amount": 200,
            "category": 4,

```

```

    "date": "08.01.2018 21:32:09",
    "description": "Поход на каток",
    "id": 8
  }
]
}

```

planning-page.component.html

```

<div class="title-block">
  <h3 class="title">
    Страница планирования <span class="sparkline bar"></span>
  </h3>
</div>
<section class="section text-center" *ngIf="!isLoading"><app-loader></app-
loader></section>

<section class="section" *ngIf="isLoading">
  <div class="row">
    <div class="col-md-12">
      <div class="card">
        <div class="card-header card-header-sm bordered">
          <div class="header-block">
            <h3 class="title">Расходы</h3>
          </div>
          <h5 class="planning-expenses pull-right">
            Общий остаток: <span class="text-success">{{bill.value |
number:'1.2'}} Р</span>
          </h5>
        </div>
        <div class="card-block">
          <div class="row" *ngFor="let c of categories">
            <div class="col-xs-6">
              <div class="n-progress">
                <div class="progress-bar {{getCatColorClass(c)}}">
                  [ngStyle]="{width: getCatPercent(c)}">
                    <span>{{c.name}}</span>
                  </div>
                </div>
              </div>
            <div class="col-xs-6">
              <p>
                <span class="text-{{getCatColorClass(c)}}">{{
getCategoryCost(c) | number:'1.2' }}</span>
                из
                <span class="text-primary">{{ c.capacity | number:'1.2'
}}</span>
                |
                осталось <span class="text-{{getCatColorClass(c)}}">{{
c.capacity - getCategoryCost(c) | number:'1.2' }}</span> (руб.)
              </p>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>

```

planning-page.component.ts


```

import { Component, OnDestroy, OnInit } from '@angular/core';
import { BillService } from '../shared/services/bill.service';
import { CategoriesService } from '../shared/services/categories.service';
import { EventsService } from '../shared/services/events.service';
import { Observable } from 'rxjs/Observable';
import { Bill } from '../shared/models/bill.model';
import { Category } from '../shared/models/category.model';
import { APPEvent } from '../shared/models/event.model';
import { Subscription } from 'rxjs/Subscription';
import { Meta, Title } from '@angular/platform-browser';

@Component({
  selector: 'app-planning-page',
  templateUrl: './planning-page.component.html',
  styleUrls: ['./planning-page.component.scss']
})
export class PlanningPageComponent implements OnInit, OnDestroy {

  isLoading = false;
  s1: Subscription;

  bill: Bill;
  categories: Category[] = [];
  events: APPEvent[] = [];

  constructor(private billService: BillService,
               private categoriesService: CategoriesService,
               private eventsService: EventsService,
               private title: Title,
               private meta: Meta
  ) {
    title.setTitle('Планирование');
    meta.addTags([
      { name: 'keywords', content: 'планирование' },
      { name: 'description', content: 'Страница планирования' }
    ]);
  }

  ngOnInit() {
    this.s1 = Observable.combineLatest(
      this.billService.getBill(),
      this.categoriesService.getCategories(),
      this.eventsService.getEvents()
    ).subscribe((data: [Bill, Category[], APPEvent[]]) => {
      this.bill = data[0];
      this.categories = data[1].filter(e => e.capacity !== 0);
      this.events = data[2];

      this.isLoading = true;
    });
  }

  getCategoryCost(cat: Category): number {
    const catEvents = this.events.filter(e => e.category === cat.id && e.type === 'outcome');
    return catEvents.reduce((total, e) => {
      total += e.amount;
      return total;
    }, 0);
  }

  private getPercent(cat: Category): number {
    const percent = (100 * this.getCategoryCost(cat)) / cat.capacity;
    return percent > 100 ? 100 : percent;
  }

```

```

    }

    getCatPercent(cat: Category): string {
        return this.getPercent(cat) + '%';
    }

    getCatColorClass(cat: Category): string {
        const percent = this.getPercent(cat);
        return percent < 60 ? 'success' : percent >= 100 ? 'danger' : 'warning';
    }

    ngOnDestroy() {
        if (this.s1) {
            this.s1.unsubscribe();
        }
    }
}

```

bill-card.component.html

```

<div class="col col-xs-12 col-sm-12 col-md-6 col-xl-5 stats-col">
  <div class="card stats" style="height: 291px;">
    <div class="card-block">
      <div class="title-block">
        <h4 class="title">Чет</h4>
      </div>
      <div class="row row-sm stats-container">
        <div class="col-xs-12 stat-col">
          <div class="stat-icon"> <i class="fa fa-rub"></i> </div>
          <div class="stat">
            <div class="value">{{bill.value | number:'1.2'}}</div>
          </div>
          <progress class="progress stat-progress" value="100" max="100">
            <div class="progress">
              <span class="progress-bar" style="width: 100%;"></span>
            </div>
          </progress>
        </div>
        <div class="col-xs-12 stat-col">
          <div class="stat-icon"> <i class="fa fa-dollar"></i> </div>
          <div class="stat">
            <div class="value">{{dollar | number:'1.2'}}</div>
          </div>
          <progress class="progress stat-progress" value="100" max="100">
            <div class="progress">
              <span class="progress-bar" style="width: 100%;"></span>
            </div>
          </progress>
        </div>
        <div class="col-xs-12 stat-col">
          <div class="stat-icon"> <i class="fa fa-euro"></i> </div>
          <div class="stat">
            <div class="value">{{euro | number:'1.2'}}</div>
          </div>
          <progress class="progress stat-progress" value="100" max="100">
            <div class="progress">
              <span class="progress-bar" style="width: 100%;"></span>
            </div>
          </progress>
        </div>
      </div>
    </div>
  </div>
</div>

```

bill-card.component.ts

```
import { Component, Input, OnInit } from '@angular/core';
import { Bill } from '../../shared/models/bill.model';

@Component({
  selector: 'app-bill-card',
  templateUrl: './bill-card.component.html',
  styleUrls: ['./bill-card.component.scss']
})
export class BillCardComponent implements OnInit {

  @Input() bill: Bill;
  @Input() currency: any;

  dollar: number;
  euro: number;

  constructor() { }

  ngOnInit() {
    const { rates } = this.currency;
    this.dollar = rates['USD'] * this.bill.value;
    this.euro = rates['EUR'] * this.bill.value;
  }
}
```

currency-card.component.html

```
<div class="col col-xs-12 col-sm-12 col-md-6 col-xl-7 history-col">
  <div class="card">
    <div class="card-block">
      <div class="title-block">
        <h4 class="title">Курс</h4>
      </div>
      <div class="row row-sm stats-container">
        <table class="table table-hover">
          <thead>
            <tr>
              <th>Валюта</th>
              <th>Курс</th>
              <th>Дата</th>
            </tr>
          </thead>
          <tbody>
            <tr>
              <td>RUB</td>
              <td>1</td>
              <td>{{ currency.date | appMoment: 'YYYY-MM-DD' }}</td>
            </tr>
            <tr *ngFor="let c of currencies">
              <td>{{ c }}</td>
              <td>{{ currency.rates[c] }}</td>
              <td>{{ currency.date | appMoment: 'YYYY-MM-DD' }}</td>
            </tr>
          </tbody>
        </table>
      </div>
    </div>
  </div>
</div>
```

currency-card.component.ts

```
import { Component, Input } from '@angular/core';

@Component({
  selector: 'app-currency-card',
  templateUrl: './currency-card.component.html',
  styleUrls: ['./currency-card.component.scss']
})
export class CurrencyCardComponent {
  @Input() currency: any;
  currencies: string[] = ['USD', 'EUR'];
}
```

bill-page.component.html

```
<div class="title-block">
  <h3 class="title pull-left">
    Страница счёта <span class="sparkline bar"></span>
  </h3>
  <div class="pull-right">
    <button class="btn-sm btn btn-primary-outline" (click)="onRefresh()">
      <i class="fa fa-refresh"></i>
    </button>
  </div>
</div>
<section class="section">
  <div class="row text-center" *ngIf="!isLoading">
    <app-loader></app-loader>
  </div>
  <div class="row" *ngIf="isLoading">
    <app-bill-card [bill]="bill" [currency]="currency"></app-bill-card>
    <app-currency-card [currency]="currency"></app-currency-card>
  </div>
</section>
```

bill-page.component.ts

```
import { Component, OnDestroy, OnInit } from '@angular/core';
import { Observable } from 'rxjs/Observable';
import { Subscription } from 'rxjs/Subscription';

import { BillService } from '../shared/services/bill.service';
import { Bill } from '../shared/models/bill.model';
import { Meta, Title } from '@angular/platform-browser';

@Component({
  selector: 'app-bill-page',
  templateUrl: './bill-page.component.html',
  styleUrls: ['./bill-page.component.scss']
})
export class BillPageComponent implements OnInit, OnDestroy {
  sub1: Subscription;
  sub2: Subscription;

  currency: any;
  bill: Bill;

  isLoading = false;

  constructor(private billService: BillService, private title: Title, private
meta: Meta) {
    title.setTitle('Счёт');
    meta.addTags([
      { name: 'keywords', content: 'счёт' },
      { name: 'description', content: 'Страница счёта' }
    ]);
  }
}
```

```

ngOnInit() {
  this.sub1 = Observable.combineLatest(
    this.billService.getBill(),
    this.billService.getCurrency()
  ).subscribe((data: [Bill, any]) => {
    this.bill = data[0];
    this.currency = data[1];
    this.isLoading = true;
  });
}

onRefresh() {
  this.isLoading = false;
  this.sub2 = this.billService.getCurrency()
    .delay(2000)
    .subscribe((currency: any) => {
      this.currency = currency;
      this.isLoading = true;
    });
}

ngOnDestroy() {
  this.sub1.unsubscribe();
  if (this.sub2) {
    this.sub2.unsubscribe();
  }
}
}

```

bill.service.ts

```

import { Injectable } from '@angular/core';
import { Http, Response } from '@angular/http';
import { Observable } from 'rxjs/Observable';

import { Bill } from '../models/bill.model';
import { BaseApi } from '../.../shared/core/base-api';

@Injectable()
export class BillService extends BaseApi {
  constructor(public http: Http) {
    super(http);
  }

  getBill(): Observable<Bill> {
    return this.get('bill');
  }

  updateBill(bill: Bill): Observable<Bill> {
    return this.put('bill', bill);
  }

  getCurrency(base: string = 'RUB'): Observable<any> {
    return this.http.get(`http://api.fixer.io/latest?base=${base}`)
      .map((response: Response) => response.json());
  }
}

```

base-api.ts

```

import { Http, Response } from '@angular/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs/Observable';
import { environment } from '../.../environments/environment';

```

```

@Injectables()
export class BaseApi {
  private baseUrl = environment.backendUrl;

  constructor(public http: Http) {
  }

  private getUrl(url: string = ''): string {
    return this.baseUrl + url;
  }

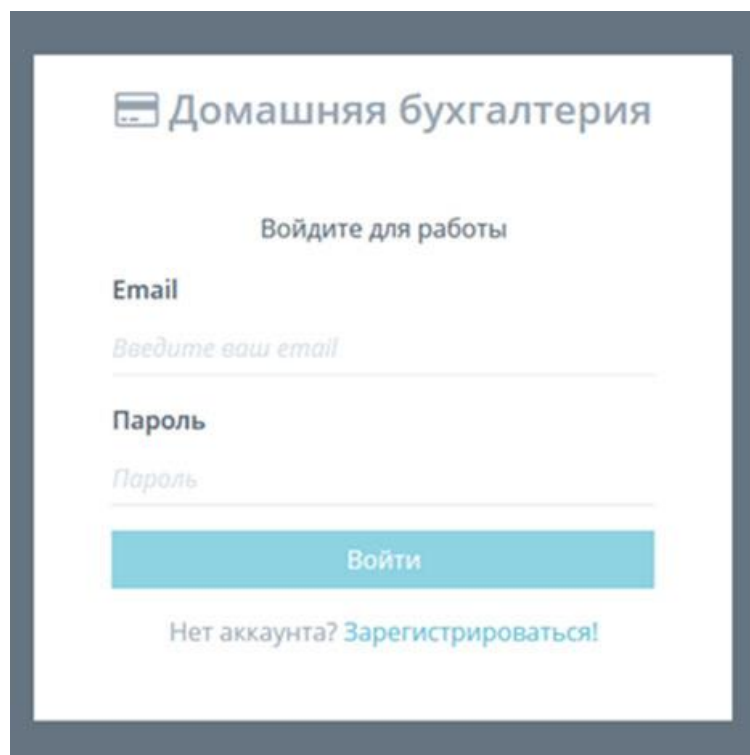
  public get(url: string = ''): Observable<any> {
    return this.http.get(this.getUrl(url))
      .map((response: Response) => response.json());
  }

  public post(url: string = '', data: any = {}): Observable<any> {
    return this.http.post(this.getUrl(url), data)
      .map((response: Response) => response.json());
  }

  public put(url: string = '', data: any = {}): Observable<any> {
    return this.http.put(this.getUrl(url), data)
      .map((response: Response) => response.json());
  }
}

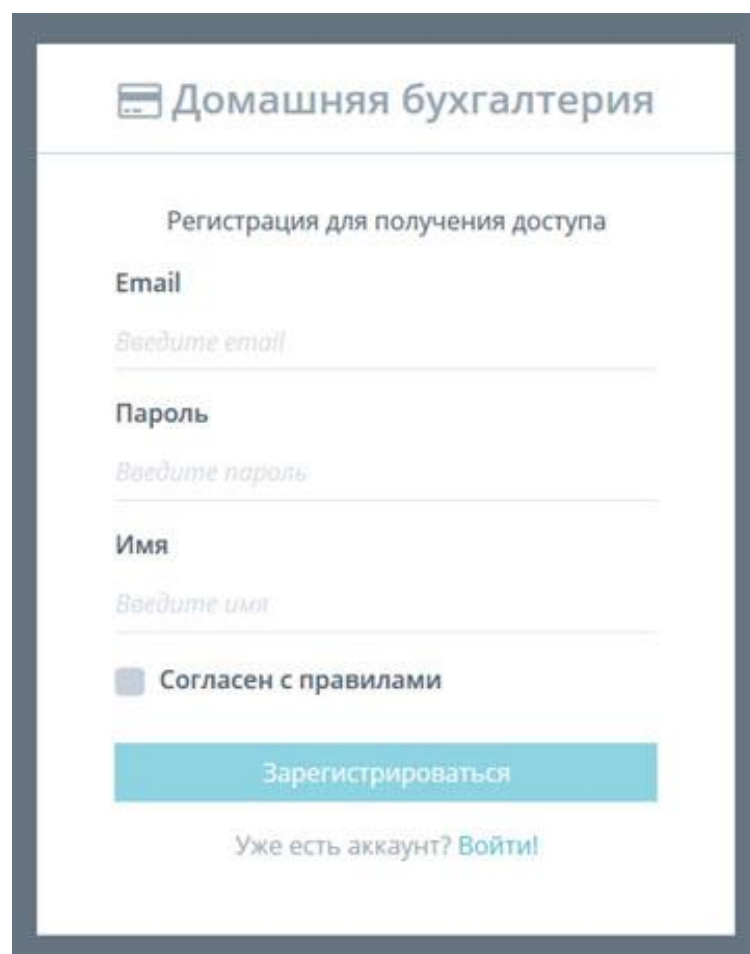
```

Результат работы



The screenshot shows a login interface for 'Домашняя бухгалтерия' (Home Accounting). At the top, there is a header with a card icon and the title. Below the header, the text 'Войдите для работы' (Log in to work) is centered. The form contains two input fields: 'Email' with a placeholder 'Введите ваш email' and 'Пароль' (Password) with a placeholder 'Пароль'. Below these fields is a blue button labeled 'Войти' (Log in). At the bottom, there is a link 'Нет аккаунта? Зарегистрироваться!' (No account? Register!).

Рисунок 1. Страница авторизации



The screenshot shows a registration interface for 'Домашняя бухгалтерия' (Home Accounting). At the top, there is a header with a card icon and the title. Below the header, the text 'Регистрация для получения доступа' (Registration for access) is centered. The form contains three input fields: 'Email' with a placeholder 'Введите email', 'Пароль' (Password) with a placeholder 'Введите пароль', and 'Имя' (Name) with a placeholder 'Введите имя'. Below these fields is a checkbox labeled 'Согласен с правилами' (Agree with terms). At the bottom, there is a blue button labeled 'Зарегистрироваться' (Register). Below the button, there is a link 'Уже есть аккаунт? Войти!' (Already have an account? Log in!).

Рисунок 2. Страница регистрации

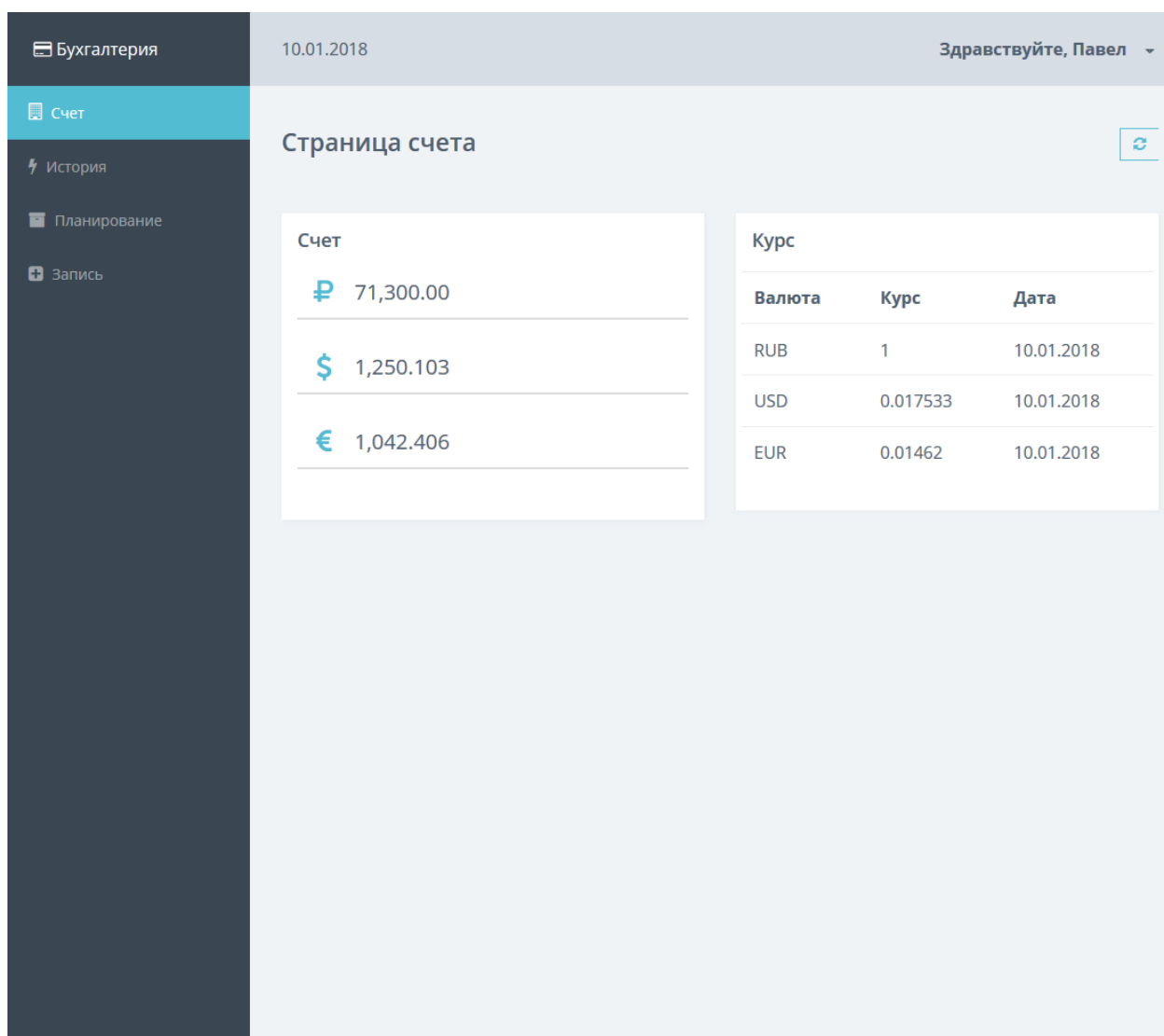


Рисунок 3. Страница счета

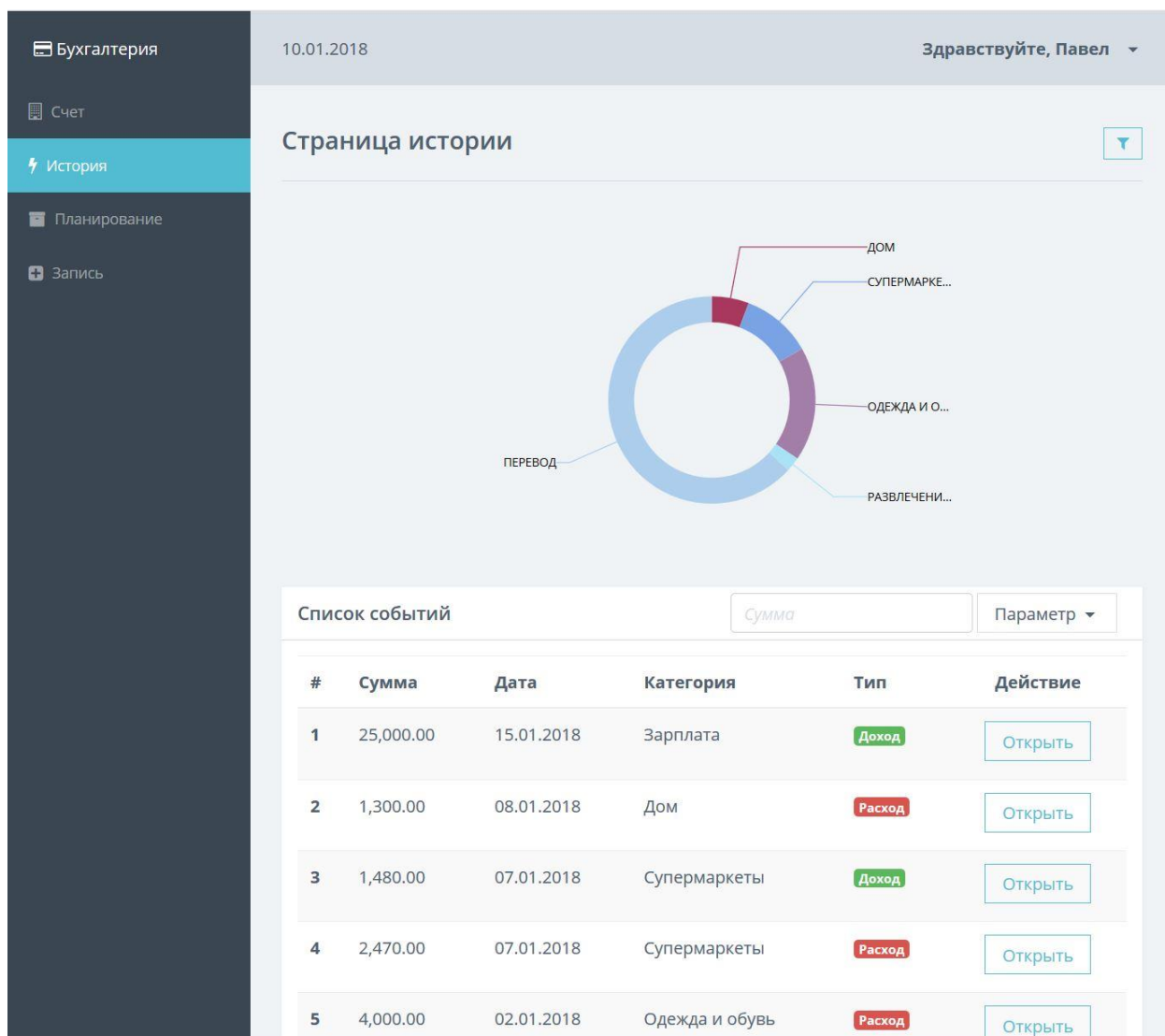


Рисунок 4. Страница истории

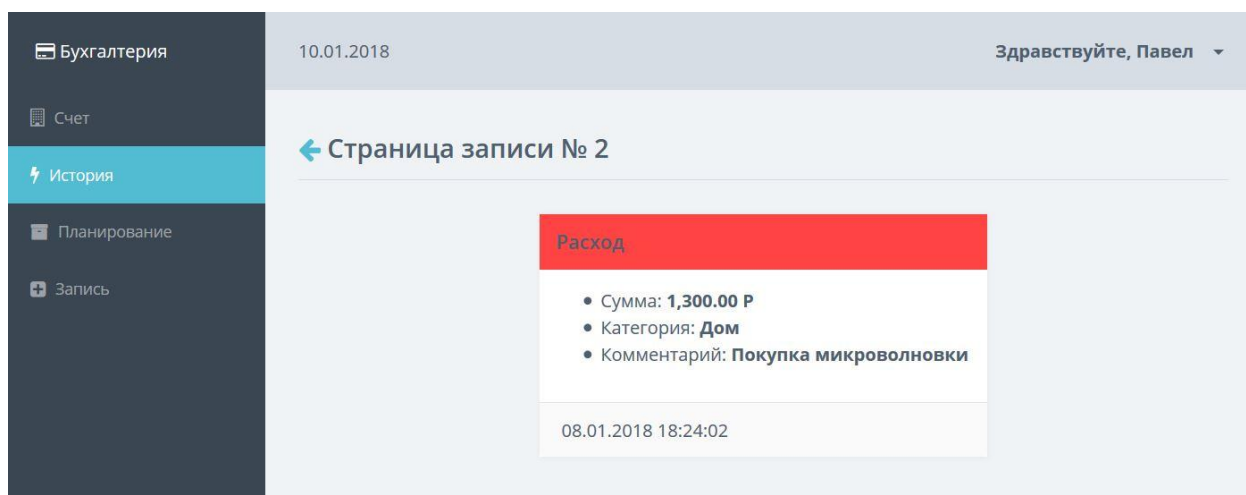


Рисунок 5. Страница с информацией по конкретной записи

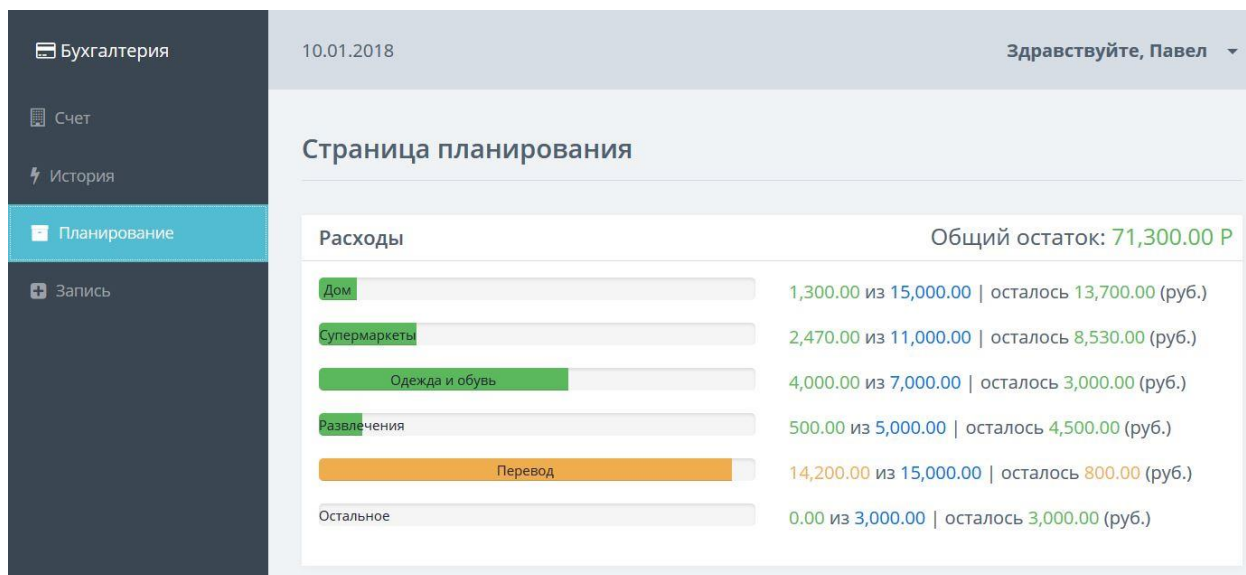


Рисунок 6. Страница планирования

Бухгалтерия

Счет

История

Планирование

Запись

10.01.2018

Здравствуйте, Павел ▾

Страница записей

Добавить событие

Выберите категорию

Дом ▾

Выберите тип

☐ Доход

☒ Расход

Введите сумму

1 ▴ ▾

Введите описание

Добавить

Редактировать категорию

Выберите категорию

Дом ▾

Введите название

Дом

Введите лимит

15000 ▴ ▾

Редактировать

Добавить категорию

Введите название

Введите лимит

1 ▴ ▾

Добавить

Рисунок 7. Страница записей