

程序设计报告

（ 2020 /2021 学年 第 一 学期）

题 目：**随手记账**

|  |  |
| --- | --- |
| **专 业** | **通信与信息工程学院**  **电子信息工程** |
| **组长 学号姓名** | **B19011722 张凌豪** |
| **组员 学号姓名** | **B19011701 于艾可** |
|  | **B1901723 陆城瑞** |
| **指 导 教 师** | **俞琼** |
| **指 导 单 位** | **计算机学院、软件学院、网络空间安全学院** |
| **日 期** | **2020.9.28** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **成员分工** | | | | | | | | | |
| 组长（张凌豪 ） | | 主要负责项目开发策划、需求分析、开发周期制定、技术栈选择、项目架构和核心模块的搭建、答辩有关事宜、教授并指导组员的学习进度和相关知识。 | | | | | | | |
| 组员（陆城瑞 ） | | 主要负责个别组件和模块的初步构建（如MainPage部分的AppBar等）、实验报告的部分撰写、学习JavaScript的React框架基本知识 | | | | | | | |
| 组员（于艾可 ） | | 主要负责UI的选择和实验报告的撰写，少数模块组件的初步搭建（如MainHeader等页面的Date组件）、思维导图制作、学习JavaScript的React框架基本知识。 | | | | | | | |
| **教师评价** | | | | | | | | | |
| 评价准则 | | | | 优秀 | 良好 | | 中等 | 合格 | 不合格 |
| 该课题组掌握程序设计开发的相关工程基础知识，并能够针对求解的工程问题，进行合理的分析与综合。 | | | |  |  | |  |  |  |
| 该课题组能够结合计算机软硬件资源，合理选用算法、数据结构、数据存储方式等技术手段，对求解的工程问题进行系统设计、建模及预测。 | | | |  |  | |  |  |  |
| 该课题组能够选择合适的程序设计语言与编程开发平台，对求解的工程问题进行系统模拟、仿真与预测。 | | | |  |  | |  |  |  |
| 该课题组熟练掌握调试方法与工具，对程序开发过程中出现的问题进行分析、跟踪与调试，并在开发平台上对最终作品进行充分测试。 | | | |  |  | |  |  |  |
| 该课题组具备初步的开发能力，完成了课题要求的各项开发任务，功能实现全面。 | | | |  |  | |  |  |  |
| 该课题组具备初步的界面设计意识。用户使用界面布局合理，交互友好，操作简便。 | | | |  |  | |  |  |  |
| 该课题组正确、完整的回答了指导教师关于课题作品的提问，反映出其对课题内容，以及相关的工程基础知识具有较好的理解和掌握。 | | | |  |  | |  |  |  |
| 该课题组具备一定的表达能力与文字处理能力，能够熟练使用文字处理软件完成课题报告一份。报告完整记录了课题组的工作及总结，内容详实，格式规范。 | | | |  |  | |  |  |  |
| 该课题组具有较好的团队协作精神，能够努力沟通解决遇到的各种开发、工程问题。 | | | |  |  | |  |  |  |
| 该课题组具备一定自学能力与探索创新意识，能够充分利用各种网络与图书馆资源自学新知识与新技能。 | | | |  |  | |  |  |  |
| 该课题组在上机过程中遵守机房规章制度，出勤与平时表现规范。 | | | |  |  | |  |  |  |
| 最终作品和报告反映出该课题组具备较好的解决工程问题的能力。 | | | |  |  | |  |  |  |
| **本次程序设计能力达成评价** | | | | | | | | | |
| 学号姓名 | B19011722 张凌豪 | | B19011723 陆城瑞 | | | B1901701 于艾可 | | | |
| 贡献度及个人表现 | 50% | | 25% | | | 25% | | | |
| 最终成绩 |  | |  | | |  | | | |
| **备注：**  贡献度及个人表现：好、较好、中等、较差、差  最终成绩：优秀、良好、中等、及格、不及格 | | | | | | | | | |
| **指导教师： 俞琼 2020 年 9 月 30 日** | | | | | | | | | |

**随手记账**

**一、课题内容和要求**

你是否会管理自己的财务呢？每天记账一小步，财务自由一大步，每天坚持记账，方便查账，做好自己的消费管理吧，请完成一个记账小程序，主要功能包括帐务收支明细数据的录入、查找、修改、删除、统计、排序等。所有帐务数据都要利用文件系统保存，以备系统下次运行时使用。通过此课题，熟练掌握文件、数组、指针的各种操作，以及一些基本算法思想的应用。

具体功能要求

(1) 有明确的多级菜单选择功能。

(2) 能够随时添加帐务明细数据。

(3) 查找帐务数据（可按序号、日期（年、月、日）、名称、金额等查找）。

(4) 修改帐务明细条目（先查找，再修改。若查找出多个条目，则进一步提示用户选择条目的序号，再修改）。

(5) 删除帐务明细条目（请参考上面修改的处理）。

(6) 对帐务数据排序（可按日期、名称、金额等排序。要求排序后，记录的序号也要重新编排）。

(7) 显示当前所有帐务数据列表。

(8) 帐务统计（根据日期范围，可分别按年、月或日统计收、支、结余总金额，并显示）

(9) 保存帐务数据到文件中、从文件中读入帐务数据。

选做要求：

（1）对用户的输入进行一定的正确性检查。

（2）涉及到日期类型数据，一致以2017/10/01形式存放，生成月报表

（3）设计一个柱形图，显示最近3个月的消费统计图。

其他要求：

（1）界面美观，交互方便。

（2）注释详细：每个变量都要求有注释说明用途；函数有注释说明功能，对参数、返回值也要以注释的形式说明用途；关键的语句段要求有注—释解释。

（3）程序的层次清晰，可读性强。

（4）变量、函数命名符合规范。

（5）如有可能，可使用MFC 等开发工具，实现彩色或图形操作界面。

**二、需求分析**

随手记账系统的功能框架图如图1所示。

图示

描述已自动生成

图1 功能框架图

随手记账的程序设计目的：方便账目的统计查询与修改，方便生活，同时警示消费，改善消费理念。

随手记账的功能：包括帐务收支明细数据的录入、查找、修改、删除、统计、排序等。

随手记账程序要求：力求界面美观、可操作性强、跨平台兼容性好、能够对账单数据进行存储和导入，以便于多次使用。

具体而言：（根据图1功能框架图）

MiniDrawer：侧边菜单栏（导航栏）——实现对各个页面的导航

MainPage：首页账单页——首页展现账单明细列表，同时显示增加按钮

SearchPage：账单筛选页——对账单明细进行条件筛选，可以按照账目的收入/支出类型、金额大小、时间等条件进行检索。

AnlazePage: 统计分析页——能够对账目进行统计分析，以图表形式展现出来

Community：社区页——能够进行社区分享

SettingPage：设置页——对程序进行一些基本的设置，同时能够对账单数据进行导入和导出。

关于数据存储：

程序运行生命周期内能够实现快捷的数据存储和读写，同时在程序生命周期结束时，能够对数据进行导出，以便于下次运行时导入数据。

**三、概要设计**

1 技术选型

根据需求分析可知，需要有一个前台页面，同时能够具有对数据操作。

将采用Web应用的前后端分离的开发模式。

因为本应用为轻量级程序，故此后端将不另外开发部署，数据操作将使用LocalStorage缓存才实现数据的操作。

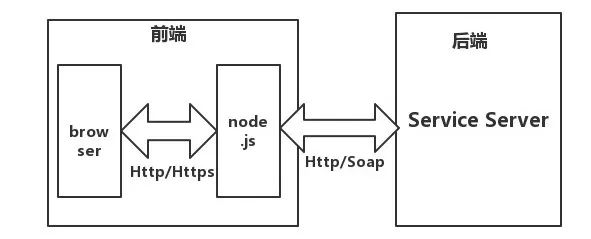


图2 前后端分离

**技术栈：**

React+Electron开发框架，使用Material-UI作为UI组件。

**为什么选择Web开发？**

经过多年的web跨端技术发展，Web App已经成为主流的程序开发选项，同时，前端技术衍生出来的PC桌面端开发框架也日益成熟，有不少优秀的项目已经投入生产。因此，我们选择使用Web技术来开发这个记账系统。同时，web开发将有更高的可复用性、可扩展性、可移植性等特点，有利于程序的多端开发和推广。

**为什么选择React+Electron？**

我们注意到程序中会有很多可以复用的模块和组件（类比于C++中的类），我们需要一个技术栈能够提供比较好的模块化、工程化的开发框架，赋予我们能快速构建一个抽象、可复用的模块化开发框架。

而React便是这么一个组件化开发框架。另外，经过多年的技术沉淀，react框架的组件库已经非常丰富和完善，如material-UI库提供了大量的组件模块，可以直接引入并加以组合构建。同时，react基于MVC设计模式所发展出来的MVVM模式，能够对View层面进行差量更新的渲染，使得网页渲染性能有所提高，我们只需要对JSX进行修改，便可以更新页面渲染。最后，react所发展出来的一些框架，例如Rax等，有着较好的多端移植能力，后期可以实现一套代码，多端适配，可以开发出小程序、app等形式。

而Electron是基于JavaScript的PC桌面端应用开发框架，最著名的成果是代码编辑器VScode，它与React的结合使得我们对桌面端应用用着高效的开发效率。

2 数据存储方式

我们使用LocalStorage存储数据，数据格式使用Json。

以下是数据存储结构：

|  |  |
| --- | --- |
| Key | Value |
| IncomeType | 收入类型 |
| ExpenseTYpe | 支出类型 |
| Data | 账目数据 |
| Tot | 支出/收入总计 |
| Days | 时间戳 |

其中，data部分存储具体的账目信息，结构如下：

|  |  |
| --- | --- |
| Key | Value |
| No | ID编号 |
| Content | 账单备注/内容 |
| Date | 时间 |
| Money | 账单数目 |
| Type | 账单类型 |

以下是一条账单的样例：

{

content: "123"

date: "2020-12-13T16:00:00.000Z"

money: -213

no: 7

type: "蔬菜"}

关于数据导出：

程序将具备对LocalStorage中json格式数据导出成excel表格的功能：

将使用react中的XLSX组件实现：XLSX.utils.json\_to\_sheet(sheetData1)

3 主要模块实现

1. LocalStorage的读写删改：

使用React的Storage组件，能够对LocalStorage进行操作，在具体的业务场景实例中引入（import）该组件，传递具体操作和数据参数以进行数据的更改。

如：import Storage from "./Storage";

图示

描述已自动生成

图3 数据更新模块

1. 数据筛选：

能够对账单数据进行不同方式的筛选和排序，以满足对账单的直观统计。

程序主要完成三种类型的筛选，可叠加：

按类型筛选、按时间段筛选、按金额大小筛选。

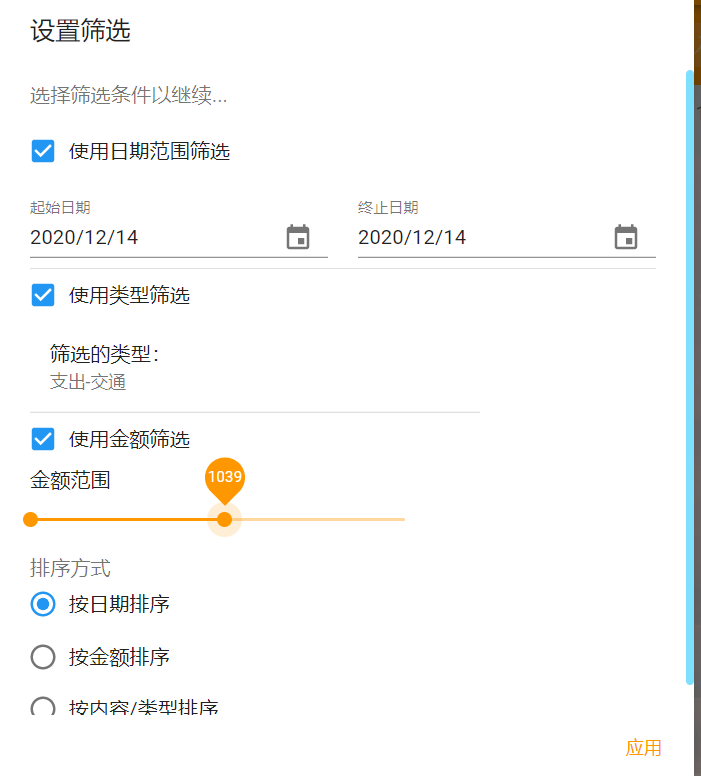


图4 账单筛选图

以模块化、组件化思维，使用组件多层封装技术，对每一种筛选方式进行统计，最后

**四、源程序代码**

目录架构：

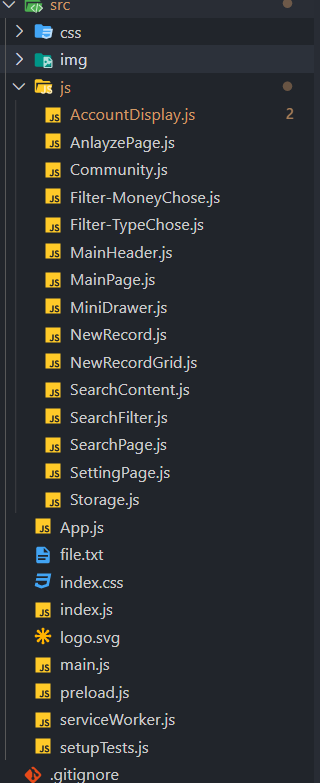


图5 项目结构图

仅展示部分文件和核心代码模块：

**Index.js——React入口文件：**

// React入口JS

ReactDOM.render(

<React.StrictMode>

<App />

</React.StrictMode>,

document.getElementById('root')

);

// React入口文件，</React.StrictMode>表示使用严格模式，这对于开发过程中的变量类型、作用域做了严格的规定，以防止组件间产生参数污染。

// <App />为自定义组件，指向App,js

**main.js——Electron入口文件：**

function createWindow() {

// 隐藏菜单栏

Menu.setApplicationMenu(null);

// 创建窗口

const mainWindow = new BrowserWindow({

width: 800,

height: 600,

frame: false,

webPreferences: {

nodeIntegration: true,

preload: path.join(\_\_dirname, 'preload.js')

}

})

// 窗口最小化、最大化、关闭

ipcMain.on('minScreen', e => mainWindow.minimize());

ipcMain.on('maxScreen', e => {

if (mainWindow.isMaximized() || mainWindow.isFullScreen()) {

mainWindow.unmaximize();

} else {

mainWindow.maximize();

}

});

}

// createWindow()方法为Electron渲染主页面进程。

**App.js——项目入口文件：**

function App() {

return (

<div className="App">

<MiniDrawer />

</div>

);

}

export default App;

// <MiniDrawer />为自定义组件，指向MiniDrawer.js，即程序内核最先链接的是菜单栏组件，也即程序左侧响应式左侧迷你变体抽屉组件

// 注意，在react中export default xxx表示对此页面组件进行封装，以供其他页面调用。

**下面简单展现react+material+electron的组件页面基本架构：**

// import相关组件

Import…

// 组件style配置

const useStyles = makeStyles({

root: {

},

…

});

// 封装导出组件XXX

export default function XXX() {

const classes = useStyles(); // 应用style配置

…

// 具体业务逻辑

…

// 组件返回JSX内容，也即组件渲染显示的代码部分

return (

<div className={classes.root}>

<xxxxx />

</div>

);

}

**MiniDrawer.js——程序左侧菜单栏：**

// 响应式左侧迷你变体抽屉组件

{/\* 顶部固定App应用栏 \*/ }

<AppBar>

<Toolbar variant="dense" className={classes.menuBar}>

<IconButton>

</IconButton>

{/\* drawer 标题栏\*/}

<Typography className={classes.title} align="left" noWrap>

C-AccountBook

</Typography>

{/\* 窗口右上最小化、最大化、关闭图标\*/}

<IconButton onClick={minAppWindow}>

</IconButton>

<IconButton onClick={maxAppWindow}>

</IconButton>

<IconButton lassName={classes.closeButton} onClick={closeAppWindow}>

</IconButton>

</Toolbar>

</AppBar >

{/\* 抽屉内容 \*/ }

<List>

<ListItem button key='账单' onClick={() => { setChose(0); }}>

<ListItemText primary={'账单'} secondary="记一笔" />

</ListItem>

<Divider />

<ListItem button key='筛选' onClick={() => { setChose(1); }}>

<ListItemText primary={'筛选'} secondary="账单明细筛选" />

</ListItem>

<Divider />

<ListItem button key='统计' onClick={() => { setChose(2); }}>

<ListItemText primary={'统计'} secondary="账单明细统计分析" />

</ListItem>

<Divider />

<ListItem button key='社区' onClick={() => { setChose(3); }}>

<ListItemText primary={'社区'} secondary="有趣的发现" />

</ListItem>

<Divider />

<ListItem button key='设置' onClick={() => { setChose(4); }}>

<ListItemText primary={'设置'} secondary="Settings~" />

</ListItem>

<Divider />

</List>

{/\* 主窗口内容

0：MainPage: 账单主页

1：SearchPage: 账单明细条件筛选

2：AnlayzePage: 账单明细统计分析

3：Community: 社区（开发ing）

4：ettingPage: 相关设置

【使用三目运算符切换】

\*/}

< main className={classes.content} >

<div className={classes.toolbar} />

{

chose == 0 ? <MainPage /> :

chose == 1 ? <SearchPage /> :

chose == 2 ? <AnlayzePage /> :

chose == 3 ? <Community /> :

<SettingPage />

}

// 组件最后有5个子页面的选择，从这里引入到之后具体的页面中去



图6 左侧响应式菜单图

**MainPage.js——账单主页：**

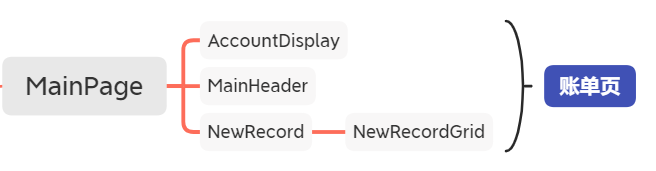


图7 账单主页图

// 这里先展示一部分组件import部分：

// 账单页（抽屉1）-首页

import React from 'react';

// 引入material-ui

import { makeStyles } from '@material-ui/core/styles';

// CSS Baseline 基线

import CssBaseline from '@material-ui/core/CssBaseline';

// Material Design 响应式布局的栅格（Grid）可适应屏幕大小和方向，确保布局在不同尺寸之间的一致性。

import Grid from '@material-ui/core/Grid';

// 账单明细显示组件

import AccountDisplay from './AccountDisplay';

// 页面投头部组件

import MainHeader from './MainHeader';

// 新增账单组件

import NewRecord from './NewRecord';

// 自适应布局模块

// https://material-ui.com/zh/components/grid/

const useStyles = makeStyles((theme) => ({

root: {

},

paper: {

},

}));

// 封装导出组件MainPage

export default function MainPage() {

const classes = useStyles();

return (

<div className={classes.root}>

{/\* 栅格式布局Grid容器 \*/}

<Grid container>

<Grid item xs={6}>

<MainHeader />

<AccountDisplay open={handleChange} />

</Grid>

<CssBaseline />

{/\* 新增账单 \*/}

<Grid item xs={6}>

<NewRecord open={checked} apply={setChecked} />

</Grid>

</Grid>

</div>

);

}

// 可以看出，MainPage包含三个组件：<MainHeader /><AccountDisplay><NewRecord>分别对应了账单主页头部栏、账单明细列表、和一个新增账单的按钮。

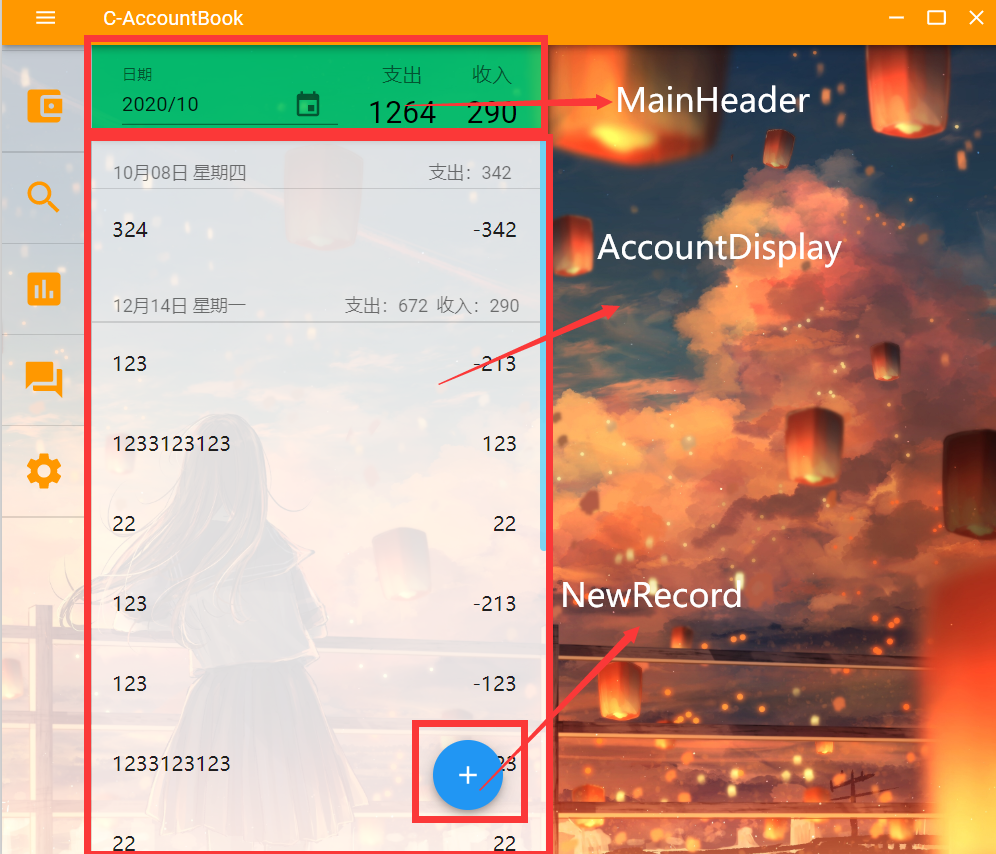


图8 MainPage组件图

**MainHeader.js——账单主页header：**

// Main账单页Header部分组件

import React from 'react';

import Box from '@material-ui/core/Box';

// App Bar 应用栏组件

import AppBar from '@material-ui/core/AppBar';

import Toolbar from '@material-ui/core/Toolbar';

import Typography from '@material-ui/core/Typography';

import { fade, makeStyles } from '@material-ui/core/styles';

import { createMuiTheme } from '@material-ui/core/styles';

import { ThemeProvider } from '@material-ui/styles';

// 日期组件

import 'date-fns';

import DateFnsUtils from '@date-io/date-fns';

import {

MuiPickersUtilsProvider,

KeyboardDatePicker,

} from '@material-ui/pickers';

// 封装导出组件MainHeader

export default function MainHeader() {

const classes = usezStyles();

return (

// 简单的应用栏组件部分

<AppBar position="static" color='primary' className={classes.bar} >

<ThemeProvider theme={datePickerTheme}>

{/\* 日期部分组件 \*/}

<MuiPickersUtilsProvider utils={DateFnsUtils}>

<KeyboardDatePicker

margin="normal"

id="date-picker-dialog"

label="日期"

format="yyyy/MM"

value={selectedDate}

onChange={handleDateChange}

KeyboardButtonProps={{

'aria-label': 'change date',

}}

/>

</MuiPickersUtilsProvider>

</ThemeProvider>

{/\* 右侧支出收入部分 \*/}

<Typography className={classes.void} />

<Box className={classes.money}>

<Typography className={classes.title} noWrap>支出<br /></Typography>

<Typography className={classes.moneyValue} noWrap>

{tot[0]}

</Typography>

</Box>

<Box className={classes.money}>

<Typography className={classes.title} noWrap>收入<br /></Typography>

<Typography className={classes.moneyValue} noWrap>

{tot[1]}

</Typography>

</Box>

</Toolbar>

</AppBar >

);

}

// 组件效果呈现如下：



图9 MainHeader组件图

**AccountDisplay.js——账单主页header：**

// 封装导出组件AccountDisplay

export default function AccountDisplay(props) {

// localStorage读

// render更新

// 模态框

// 取消修改

const handleRecordClose = () => {

setOpenModal(false);

};

// 删除数据

const handleDelRecord = () => {

Storage.delRecord(nowe);

setOpenModal(false);

};

// 修改数据

const handleNewRecordClose = () => {

Storage.modifyRecord(nowe, ALData[nowe].no, selectedDate, content, typeOfNew, money);

handleRecordClose();

};

const handleDateChange = (date) => {

setSelectedDate(date);

};

return (

<Box style={{}}>

<List className={classes.root} subheader={<li />}>

// 通过map进行日期遍历

Days.map((items) => { }

let dt = new Date(items[0]);

let getCh = ['日', '一', '二', '三', '四', '五', '六']

return (

)

</List >

<Fab>

<AddIcon />

</Fab>

{/\* 对话框/模态框 \*/}

<Dialog open={openModal} onClose={handleRecordClose} >

<DialogTitle >修改</DialogTitle>

<DialogContent>

<div style={{ display: 'inline' }}>

<MuiPickersUtilsProvider>

<KeyboardDatePicker

label="日期"

/>

</MuiPickersUtilsProvider>

<TextField

label="金额"

/>

</div>

<TextField

label="账单备注"

/>

</DialogContent>

<DialogActions>

<Button onClick={handleDelRecord} color="#f44336">

删除

</Button>

<Button onClick={handleRecordClose} color="primary">

取消

</Button>

<Button onClick={handleNewRecordClose} color="primary">

确定

</Button>

</DialogActions>

</Dialog>

</Box >

}



图10 AccountDisplay组件图

其他页面不再一一展示。

**五、测试数据及其结果分析**

# 程序设计周-随手记账

### 运行

安装依赖：

```bash

npm install

```

启动服务React：

``` bash

npm start

```

启动electron调试模式：

``` bash

npm run electron-dev

```

运行后，程序的主界面如下。



图11 主界面图

左侧为主界面，右侧为开发者工具（可关闭）

1. 添加一笔账单

数据： 餐饮类，2020/10/10，支出¥20，账单备注：吃午饭

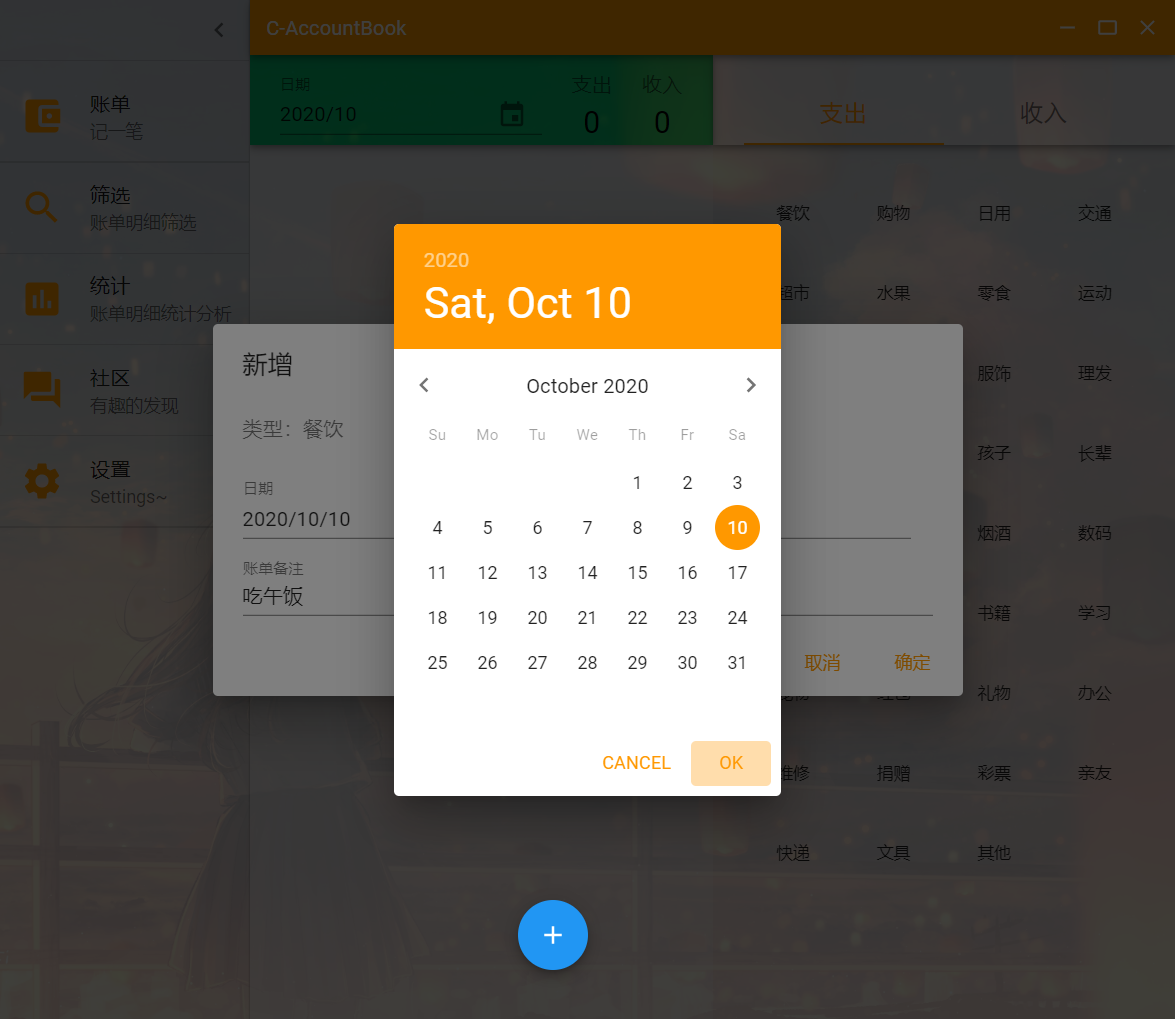


图12 添加记录图

以下为添加账单记录后主界面：



图13 添加记录后主界面图

1. 账单筛选

原始数据如下：



图14 账单筛选原始数据图

账单筛选界面：我们选择时间筛选+金额筛选

目前已筛选条件: 日期范围:2020/10/01~2020/10/16; 金额范围:0~336; (排序方式:按日期排序)

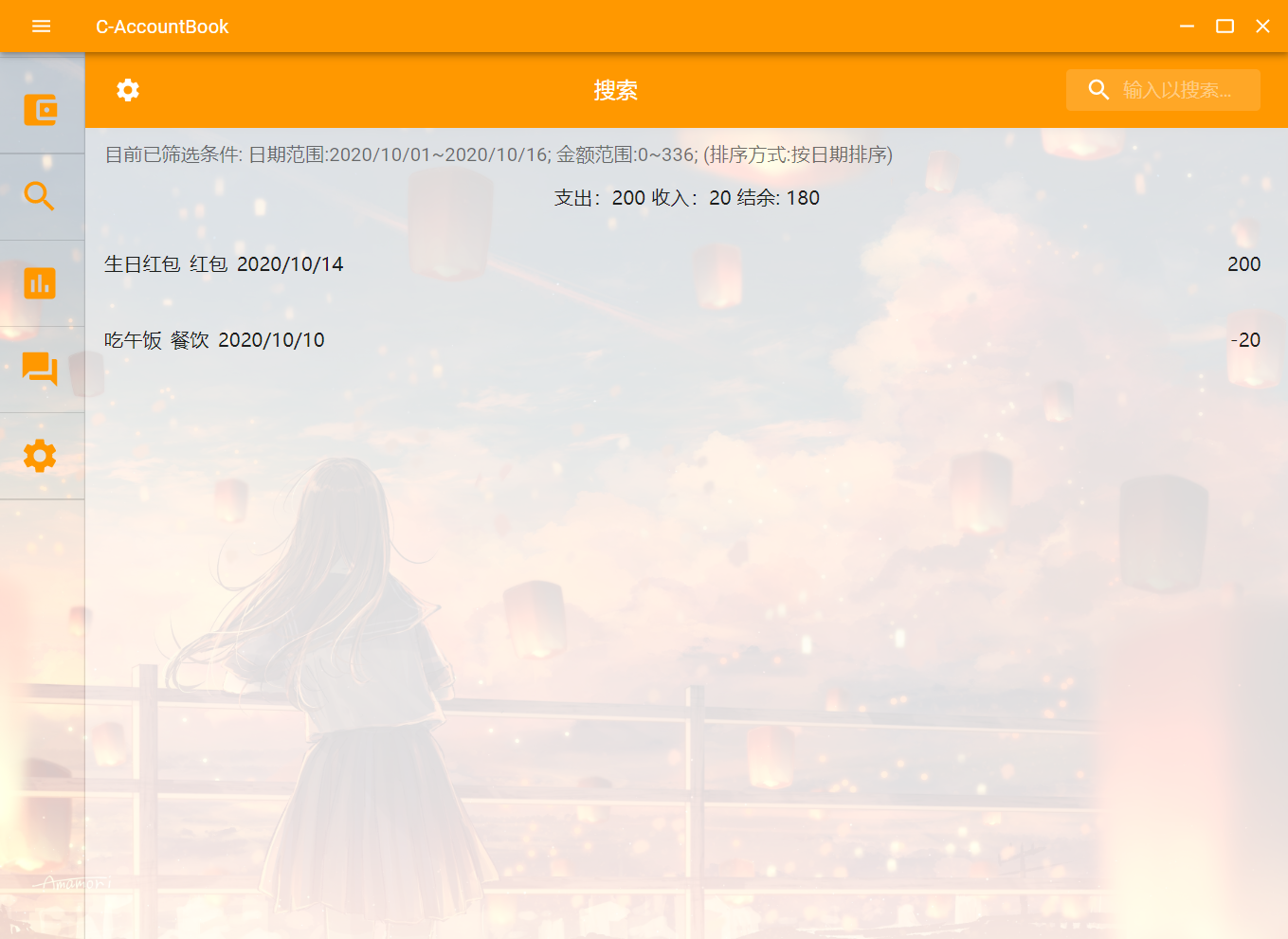


图15 账单筛选后数据图

当排序方式选择按照金额大小排序所有的账单：



图16 按照金额排序所有账单图

1. 数据导出

点击settingPage：选择导出按钮

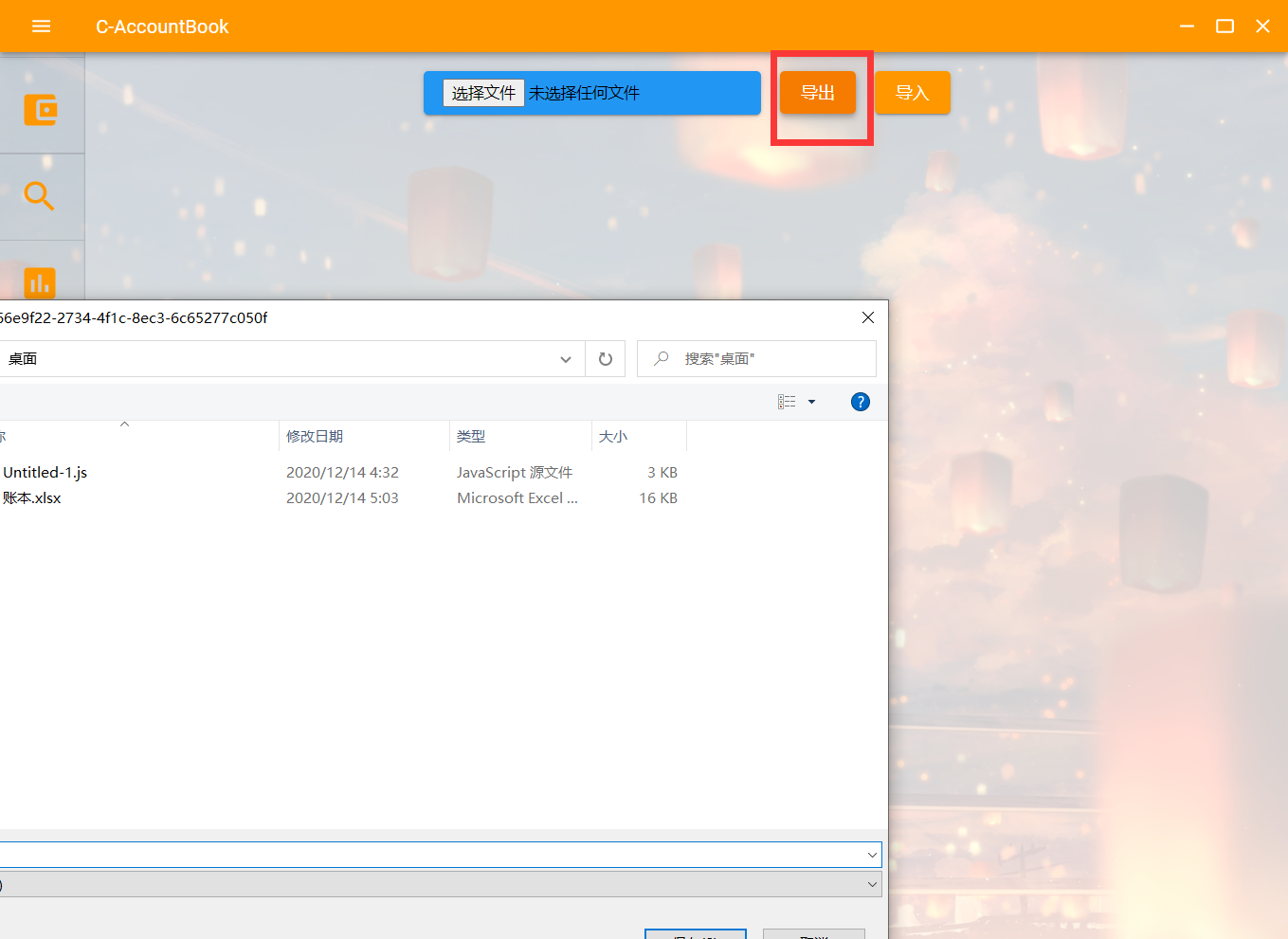


图17 数据导出图

导出结果进入Excel电子表格：

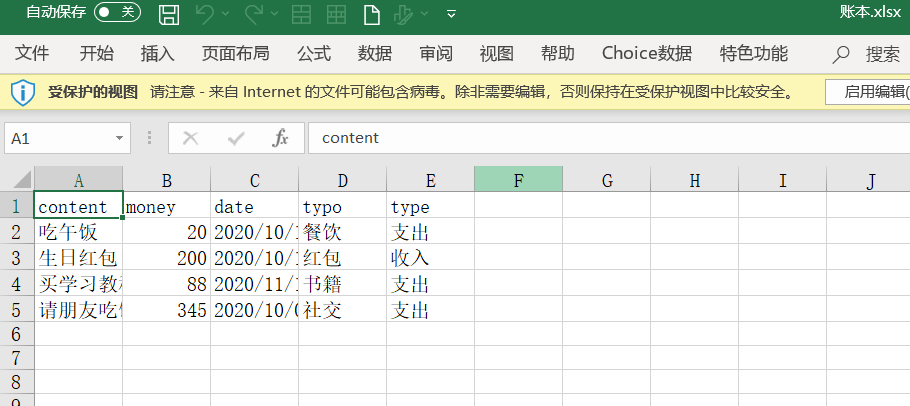


图18 导出Excel数据图

**六、调试过程中的问题**

1. 功能未全部完善：社区、数据统计功能暂未开发。
2. 当添加数据过多，尤其是通过导入excel的方式导入数据，会发生程序卡住的现象，这大概是因为LocalStorage的存储空间大约只有5MB，同时组件通信占用了大量的存储空间，导致卡顿。

......

**七、课程总结**

张凌豪：

作为SAST前端组讲师，对于web方面的授课本就是本职工作，因此我选择使用web的技术栈来实现这个程序也是顺其自然的。但确实作为一个团队的话，互相的协作是比较重要的。正是因为课程开发的时间周期并不是非常短，让我有机会和两位组员一起共同探讨学习Web开发。起初我们选择微信小程序为技术栈，也开发了一个小的成品出来，但是最终因为云函数和云数据库当时的限制最终还是回归了以react+electron为框架的PC桌面Web App的技术栈上。在这个过程中，互相的学习和探讨的过程也是对我授课的一次检验，一次锻炼。

与此同时，虽然曾经参与过React+Electron的项目开发，但是完全有自己去主导、去策划倒也还是第一次，这次经历也让我发现了自己的一些不足，主要是对于一些代码细节有些推敲。这告诫了我今后的学习和授课中也需要将目光聚集到一些具体的业务实现代码模块上，而不仅仅是框架本身和架构的开发。同时，这次开发过程也让我对于React Hooks、Material-UI有了更深入地了解，也对MVVM设计模式有了新的深刻认识。

我的两位组员虽然以前没有接触过应用开发，不管是纯粹的C++开发还是Java工程、Web前后端项目，但是在这一次程序设计过程中，也能逐步地跟上并大致理解项目开发的基本流程和模块化、组件化的设计思想，能够自己模仿着样子初步搭建一个组件的基本结构，这一点我认为是可喜的。我们在这个过程中间充分展现了研究精神。

最后，对于这一次程序设计中那些美中不足的地方，例如因为我个人平时较忙而差点忘记的课程报告，还有程序中遗留下来的个别Bug和一些值得优化的地方，我认为还是需要继续优化和处理，希望在今后的应用开发和学习生活中我能够从这样一次课程设计中得到一些经验和感悟。

于艾可：

此次的程序设计，让我深刻的感受到，在一个程序诞生之前，思维导图和框架这些看起来无关紧要的东西，有着巨大的作用。它避免了去重复去做一段代码，也能让我们在没头绪的时候，有一个思路去想从其他方面入手。

还让我认识到了团队合作的重要性，一个人的知识储备是有限的，一个人的精力也是有限的，但当几个人组在一起的时候，会将我们每个人的能力最大化。就像是从未设计过程序的我，在组员的帮助下能逐步地跟上并大致理解项目开发的 基本流程和模块化、组件化的设计思想，能够自己模仿着样子初步搭建一个组件的基本结构，对程序设计有了更深的理解。

总体下来，这次程序设计的结果都令人满意，对于我也是一次不小的收获，不仅培养了我把知识用于实践的能力，也让我接触到了各种各样的新知识，让我感受到了程序设计语言的魅力，最后我希望能有更多的机会去参与到团队合作编写程序的活动中，不断的提升自己，也很感谢我的组员给予我的帮助。

陆城瑞：

在这次程序设计任务中，有幸能够与组长和队友一起学习新的知识，我们的程序因为综合考虑下来的难易程度与程序设计周期的长度，综合考虑下来我们最终决定以学习的态度，学习web的新知识并将其付诸于实践。期初，这对于我们来说是一个不小的挑战，因为作为web小白，很难在一开始就接受并理解它，更多的是机械性的套用，这成为了我们开始工作效率低下的一个原因，不过好在在组长的带领下，我们一次一次作出新的尝试，并尝试去理解它的用法，尽管失败，bug报错经常发生，但是我们小组总是互相帮助，有不懂的地方最后都会集众智将它解决，这对于首次开发程序的我来说，确实是一次变革，也让我感受到了团队的力量与强大。

总体下来，这次程序设计的结果都令人满意，对于我也是一次不小的收获，不仅培养了我把知识用于实践的能力，也让我接触到了各种各样的新知识，让我感受到了程序设计语言的魅力，也收获到了不小的成就感，更让我学会了团队之间的协作与信任。感受到了组长一直带领着我们解决难题，教会我们新的技能的组内和谐气氛。

最后，虽然经过三个人的努力，程序还是有美中不足的地方，尽管在组长的带领下，我们对这门新语言的理解也还是有不周到的地方，虽然最后也不能完全熟悉的掌握，只能机械性的套用与组合，但最后还是受益匪浅，学会了许多课外拓展的新知识。通过这次任务，终究还是体验到了程序员的艰辛，也对程序设计刚开始的害怕逐渐学会了去逐步分析解决问题，从刚开始的不知所措变得得心应手，我的设计思维与能力也得到了很大的提升。