**《智能移动开发》结课作业**

姓名：向直柳 学号：2014600 班级：2020级二学位

作业小程序AppID：wx000399b1cfd6c6eb

**1. 作业任务**

**1.1 选题描述：便民就医工具**

**1.1.1 需求背景**

当前农村三级卫生服务网（以县级卫生医疗机构为龙头，乡镇卫生院为主体，村卫生室为基础的卫生服务体系）资源匮乏、卫生人力资源再生性困难、卫生服务能力严重不足等结构性难题仍然存在。面对这些问题，需要充分利用"人工智能诊疗系统"，使得"远程诊疗"成为可能，并利用人工智能加强农村居民的自我健康管理。希望有"移动医疗"和"人工智能助力医疗模式"类的小程序工具为乡村医生提供强有力的智能助手；以及帮助乡村医生职能转型，电子档案管理和健康教育等农村公共卫生服务的工具。

**1.1.2 主要功能**

* 支持帐号登录体系，帐号登录、登出等；
* 预约挂号，可选择医院、科室、医生，并能预约取号；
* 较轻病症咨询，可与医生进行简单交流，可传输文本、图片、视频、语音；
* 有简单管理端，供医院配置挂号相关信息；
* 有医生端（根据登录人区分），可与患者交流，传文本、图片、视频、语音。

**1.1.3 选做功能**

* 支持医院院内导航，选择目的科室，显示导航路线；
* 适老化支持，如字体适配、语音播放等；
* 科学预防知识宣传，如视频、图文的查看；
* 支持实时视频对讲；
* 支持订阅消息提醒；
* 医保查询，输入相关身份信息，验证通过后，可查询医保信息。

**1.2 作业要求**

不用这么面面俱到，这个文档只需包括三个部分：

1）应用场景分析与拟实现的功能介绍；

2）采用的技术与方法介绍；

3）产品功能展示和使用方法。

**1.3 工作计划**

第一次迭代完成：（5月24日完成）

* 数据模型架构设计：ER图、类图设计，数据库设计
* 功能1：实现帐号登录体系；
* 功能2：完成医生端的可视化部分。

第二次迭代完成：(5月31日完成)

* 功能1：完成医生端的挂号查询功能；
* 功能2：完成患者端的挂号查询与新建挂号功能；

第三次迭代完成：（6月7日完成）

* 功能1：完成挂号信息中打开医患交流对话框的功能。
* 功能2：继续完善对话部分，可传输文本、图片、视频、语音；

第四次迭代完成：（6月16日完成）

* 功能1：适老化支持，如字体适配、语音播放等；
* 功能2：有简单管理端，配置挂号相关信息，完善医生的科室信息等；
* 功能3：进行UI美化。

**2. 第一次迭代**

**2.1 任务目标**

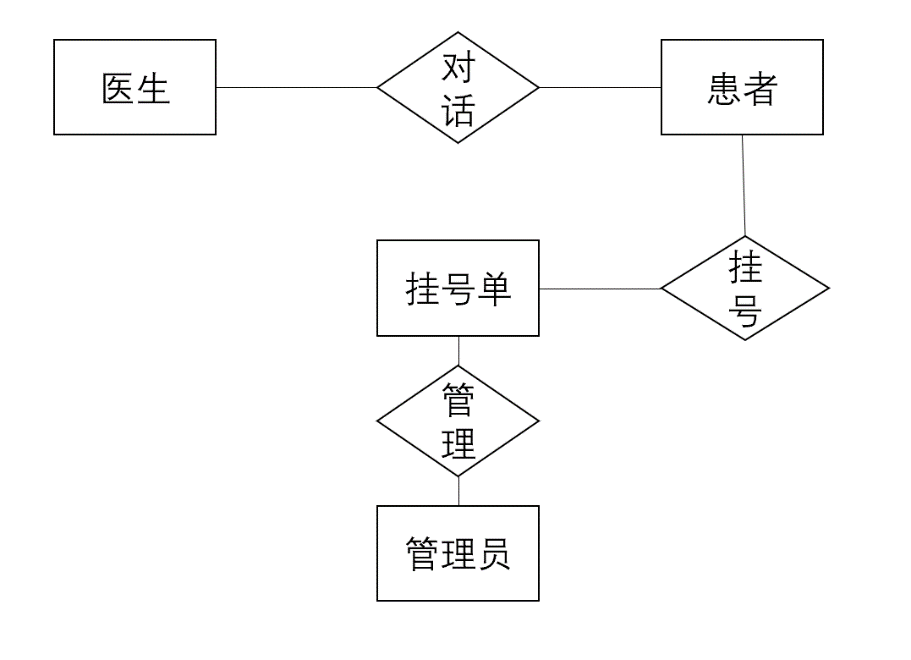
* 数据模型架构设计：实体关系图、类图设计，数据库设计
* 功能1：实现帐号登录体系；
* 功能2：完成医生端的可视化部分；

**2.2 数据模型架构的初步设计**

本节主要讨论MVC设计模式中的model部分的设计概要。

**2.2.1 实体关系设计**

主要实体包括：患者、医生、管理员；关系包括：挂号单、对话、管理，如下图所示：



**2.2.2 类抽象**

|  |
| --- |
| Class Diagram  Person {  string id  string passwd  enum type  func logup() {}  func login() {}  func logout() {}  func resetpw() {}  func delete() {}  } |
| Patient: Person {  func register() {}  func chat() {}  }  Manager: Person {  func manage\_order() {}  //reserved  } |
| Doctor: Person {  List <Order> orderlist;  func chat() {}  } |
| Order {  Patient patient  Doctor doctor  func process() {}  func create\_dialog() {}  } |
| Dialog {  Patient patient  Doctor doctor  List <String> messages  func process() {}  func create\_dialog() {}  } |

**2.2.3 数据库设计**

User表

|  |  |
| --- | --- |
| username | 用户名 |
| password | 密码 |
| type | 用户类型 |

Order表

|  |  |
| --- | --- |
| orderid | 挂号编号 |
| patient | 患者 |
| doctor | 医生 |
| date | 日期 |
| rank | 排号 |

Dialog表

|  |  |
| --- | --- |
| dialogid | 会话编号 |
| orderid | 挂号编号 |
| patient | 患者 |
| doctor | 医生 |
| messages | 聊天内容 |

**2.3 功能实现**

**2.3.1 Viewer部分**

* 用户注册

<form bindsubmit="formSubmit" bindreset="formReset">

    <text> 新用户注册 </text>

    <br/>

    <text>用户名</text>

    <input name="username"/>

    <text>姓名</text>

    <input name="nickname"/>

    <text>密码</text>

    <input name="password"/>

    <text>重复密码</text>

    <input name="pwdcheck"/>

    <text>用户身份</text>

    <radio-group name="type">

      <label><radio value="patient"/>患者</label>

      <label><radio value="doctor"/>医生</label>

      <label><radio value="manager"/>管理人员</label>

    </radio-group>

    <button formType="submit" type="primary">提交注册</button>

</form>

* 用户登录

<!--index.wxml-->

<view class="container">

  <view class="title">

    <text>便民就医工具</text>

  </view>

  <view class="userinfo">

    <form bindsubmit="login">

        <text>用户名</text> <input name="username"/>

        <text>密码</text> <input name="password"/>

        <button formType="submit" type="primary"> 登录 </button>

    </form>

    <button bindtap="logup"> 注册 </button>

  </view>

</view>

* 医生端界面

<!--pages/users/doctor.wxml-->

<text>便民就医工具：医生端</text>

欢迎，xx 医生

挂号信息：

<view class="table">

  <view class="tr">

    <view class="th"> ID </view>

    <view class="th"> 日期 </view>

    <view class="th"> 患者姓名 </view>

    <view class="th"> 会话  </view>

  </view>

  <block wx:for="{{orders}}" wx:key="{[orderid]}">

    <view class="tr bg-g" wx:if="{{index % 2 == 0}}">

      <view class="td">{{item.orderid}}</view>

      <view class="td">{{item.datetime}}</view>

      <view class="td">{{item.patient}}</view>

      <view class="td"> <button bindtap="createdialog"> 发起对话 </button> </view>

      To be done inthe next version, Thanks!

    </view>

    <view class="tr" wx:else>

      <view class="td">{{item.orderid}}</view>

      <view class="td">{{item.datetime}}</view>

      <view class="td">{{item.patient}}</view>

      <view class="td"> <button bindtap="createdialog" type="primary"> 发起对话 </button> </view>

    </view>

  </block>

</view>

**2.3.2 Controller部分**

* 用户注册

const userdb = wx.cloud.database().collection("users")

var username = ""

var nickname = ""

var password = ""

var pwdcheck = ""

var type = ""

Page({

  formSubmit: function (e) {

    username = e.detail.value.username

    nickname = e.detail.value.nickname

    password = e.detail.value.password

    pwdcheck = e.detail.value.pwdcheck

    type = e.detail.value.type

    console.log(username, nickname, password, type)

    if(password != pwdcheck) {

      console.log("密码不一致！")

    } else {

      userdb.where({username: username}).get().then(res => {

        if(res.data.length == 0) {

          console.log("注册成功！3秒后跳转到登陆界面")

          userdb.add({

            data: {username: username, nickname:nickname, password:password, type:type}

          })

        } else {

          console.log("用户名已被注册！")

        }

      })

    }

  },

})

* 用户登录

// index.js

const app = getApp()

const userdb = wx.cloud.database().collection("users")

var username = ""

var password = ""

var type = ""

Page({

  logup() {

    wx.navigateTo({

      url: '/pages/index/logup',

    })

  },

  login: function (e) {

    username = e.detail.value.username

    password = e.detail.value.password

    console.log(e.detail.value.username, e.detail.value.password)

    userdb.where({username:username}).get().then(res => {

      if(res.data.length == 0) {

        console.log("登陆失败：用户不存在")

      } else {

        if(password != res.data[0].password) {

          console.log("登录失败：密码错误")

        } else {

          type = res.data[0].type

          if(type == "doctor") {

            wx.navigateTo({

              url: '/pages/users/doctor',

            })

          } else if (type == "patient") {

            wx.navigateTo({

              url: '/pages/users/patient',

            })

          } else if (type == "manager") {

            wx.navigateTo({

              url: '/pages/users/manager',

            })

          } else {

            console.log("未知用户类型")

          }

        }

      }

    })

  },

})

* 医生端界面（待下个版本与对话部分一起完成数据库接口部分）

Page({

  data: {

    orders: [

      {"orderid": "1", "datetime":"2022/05/24 20:01", "patient": "Alice"},

      {"orderid": "2", "datetime":"2022/05/24 20:01", "patient": "Bob"},

      {"orderid": "3", "datetime":"2022/05/24 20:01", "patient": "Cindy"},

      {"orderid": "4", "datetime":"2022/05/24 20:01", "patient": "David"}

      //TBA

    ]

  },

})

**2.4 功能展示**

详见录屏。

**3. 第二次迭代**

**3.1 任务目标**

* 功能1：完成医生端的挂号查询功能；
* 功能2：完成患者端的挂号查询与新建挂号功能；

**3.2 本次迭代的设计变更**

* Order表：添加了新的属性state，即当前挂号处于激活状态（active）还是逾期状态（expired）；
* User表：分别对医生添加新的医生信息，包括科室department（string类型）和是否可挂号available（boolean类型）。

**3.3 功能实现**

**3.3.1 挂号查询**

* Viewer部分

<!--pages/users/patient.wxml-->

<text>患者端</text>

<button bindtap="createOrder" type="primary"> 新的预约 </button>

<text>当前挂号</text>

<view class="table">

  <view class="tr">

    <view class="th"> ID </view>

    <view class="th"> 医生姓名 </view>

  </view>

  <block wx:for="{{active\_orders}}" wx:key="{[doctor]}">

    <view class="tr bg-g" wx:if="{{index % 2 == 0}}">

      <view class="td">{{index}}</view>

      <view class="td">{{item.doctor}}</view>

    </view>

    <view class="tr" wx:else>

      <view class="td">{{index}}</view>

      <view class="td">{{item.doctor}}</view>

    </view>

  </block>

</view>

<text>历史挂号</text>

<view class="table">

  <view class="tr">

    <view class="th"> ID </view>

    <view class="th"> 医生姓名 </view>

  </view>

  <block wx:for="{{active\_orders}}" wx:key="{[doctor]}">

    <view class="tr bg-g" wx:if="{{index % 2 == 0}}">

      <view class="td">{{index}}</view>

      <view class="td">{{item.doctor}}</view>

    </view>

    <view class="tr" wx:else>

      <view class="td">{{index}}</view>

      <view class="td">{{item.doctor}}</view>

    </view>

  </block>

</view>

* Controller部分

// pages/users/patient.js

const orderdb = wx.cloud.database().collection("orders")

Page({

  data: {

    expired\_orders: [],

    active\_orders:[]

  },

  onLoad: function (options) {

    var \_this = this;

    orderdb.where({patient:"Alice", state:"active"}).get({

      success: res => {

        this.setData({

          active\_orders: res.data

          }

        ),

        console.log(doctors)

      }

    }),

    orderdb.where({patient:"Alice", state:"expired"}).get({

      success: res => {

        this.setData({

            expired\_orders: res.data

          }

        ),

        console.log(doctors)

      }

    })

  },

  createOrder() {

    wx.navigateTo({

      url: '/pages/functions/createOrder'

    })

  }

})

**3.3.2新建挂号**

* Viewer部分

<!--pages/functions/createOrder.wxml-->

<text>挂号信息</text>

<h1> 医生信息 </h1>

<view class="table">

  <view class="tr">

    <view class="th"> ID </view>

    <view class="th"> 医生姓名 </view>

    <view class="th"> 操作  </view>

  </view>

  <block wx:for="{{doctors}}" wx:key="{[nickname]}">

    <view class="tr bg-g" wx:if="{{index % 2 == 0}}">

      <view class="td">{{index}}</view>

      <view class="td">{{item.nickname}}</view>

      <view class="td"> <button bindtap="addOrder" type="primary"> 挂号 </button> </view>

    </view>

    <view class="tr" wx:else>

      <view class="td">{{index}}</view>

      <view class="td">{{item.nickname}}</view>

      <view class="td"> <button bindtap="addOrder" type="primary"> 挂号 </button> </view>

    </view>

  </block>

</view>

* Controller部分

// pages/functions/createOrder.js

const orderdb = wx.cloud.database().collection("orders")

const usersdb = wx.cloud.database().collection("users")

Page({

  data: {

    doctors: [ ]

  },

  onLoad: function (options) {

    var \_this = this;

    usersdb.where({type:"doctor"}).get({

      success: res => {

        this.setData({

            doctors: res.data

          }

        ),

        console.log(doctors)

      }

    })

  },

  addOrder: function (e) {

    orderdb.add({

      data: {orderdb: username, nickname:nickname, password:password, type:type}

    })

  },

})

**3.3 功能展示**

详见录屏。

**4. 第三次迭代**

**4.1 任务目标**

* 功能1：完成挂号信息中打开医患交流对话框的功能；
* 功能2：继续完善对话部分，可传输文本、图片、视频、语音。

**4.2 功能实现**

本次迭代主要完成对话功能，详细程序如下。

* Viewer部分

<!-- wxml -->

<scroll-view id='page' scroll-into-view="{{toView}}" upper-threshold="100"

  scroll-y="true" enable-back-to-top="true" class="message-list">

  <!-- 每一条消息 -->

  <view class="cu-chat" wx:for="{{chats}}" wx:key="index" id="row\_{{index}}">

    <!-- 自己发出的消息 -->

    <block wx:if="{{item.status==status}}">

      <block wx:if="{{item.showTime}}">

        <view class="datetime" style="width:100%">{{item.sendTime}}</view>

      </block>

      <view class="cu-item self"

        style="width: 750rpx; height: 120rpx; display: flex; box-sizing: border-box; left: 0rpx; top: 0rpx">

        <view class="main">

          <view class="content bg-green shadow" style="position: relative; left: 0rpx; top: 22rpx;border-radius: 10rpx">

            <text style="font-size:33rpx">{{item.textContent}}</text>

          </view>

        </view>

        <view class="cu-avatar radius center"

          style="background-image: url({{useravatar}}); width: 71rpx; height: 71rpx; display: flex; box-sizing: border-box; left: 0rpx; top: 0rpx"

          bindtap="go\_myinfo"></view>

      </view>

    </block>

     <!-- 对方发出的消息 -->

     <block wx:else>

      <block wx:if="{{item.showTime}}">

        <view class="datetime" style="width:100%">{{item.sendTime}}</view>

      </block>

      <view class="cu-item"

        style="width: 750rpx; height: 120rpx; display: flex; box-sizing: border-box; left: 0rpx; top: 0rpx">

          <view class="cu-avatar radius center"

            style="background-image: url({{match\_avatar}}); width: 71rpx; height: 71rpx; display: flex; box-sizing: border-box; left: 0rpx; top: 0rpx">

          </view>

        <view class="main">

          <view class="content bg-white shadow" style="position: relative; left: 0rpx; top: 22rpx;border-radius: 10rpx">

            <text style="font-size:33rpx">{{item.textContent}}</text>

          </view>

        </view>

      </view>

    </block>

  </view>

</scroll-view>

<!-- 回复框 -->

<view class="reply cu-bar">

    <view bindtap="talkToAdmi" ><button bindtap='sendMsg' class="contactadmi">报告</button>

</view>

  <!-- 输入框 -->

  <view class="opration-area">

    <input type="text" bindinput="getContent" value="{{textInputValue}}" maxlength="300" cursor-spacing="10"></input>

  </view>

  <!-- 发送按钮 -->

  <button class="cu-btn bg-green send" bindtap='sendMsg'>发送</button>

</view>

/\*wxss文件\*/

.message-list {

  margin-bottom: 54px;

}

/\*文本输入或语音录入\*/

.reply .opration-area {

  flex: 1;

  padding: 8px;

}

/\*回复文本框\*/

.reply input {

  background: rgb(252, 252, 252);

  height: 36px;

  border: 1px solid rgb(221, 221, 221);

  border-radius: 6px;

  padding-left: 3px;

}

/\*回复框\*/

.reply {

  display: flex;

  flex-direction: row;

  justify-content: flex-start;

  align-items: center;

  position: fixed;

  bottom: 0;

  width: 100%;

  height: 108rpx;

  border-top: 1px solid rgb(215, 215, 215);

  background: rgb(245, 245, 245);

}

/\*日期\*/

.datetime {

  font-size: 10px;

  padding: 10px 0;

  color: #999;

  text-align: center;

}

.contactadmi{

  width: 15%rpx;

  height: 64rpx;

  display: flex;

  box-sizing: border-box;

  left: 10rpx;

  top: 0rpx;

  position: relative;

  font-size: small;

  text-align: center;

  color: rgb(236, 236, 236);

  background-color: rgb(214, 87, 87);

  line-height: 64rpx;

}

.send{

  font-size:small;

  text-align: center;

  width: 15%rpx;

  height: 64rpx;

  display: flex;

  box-sizing: border-box;

  left:-10rpx;

  top: 0rpx;

  position: relative;

}

.input{

  width: 50%rpx;

  height: 64rpx;

  display: block;

  box-sizing: border-box;

}

* Controller部分

const app = getApp()

const db = wx.cloud.database()

const \_ = db.command

const chatroomCollection = db.collection("dialog")

Page({

  data: {

    status:'',

    record\_id: '',

    useravatar:'',

    match\_avatar:'',

    chats:[],

    textInputValue:''

  },

  onReady() {

    var that = this

    chatroomCollection.where({

        \_id: that.data.record\_id

      })

      .watch({

        onChange: this.onChange.bind(this),

        onError(err) {

          console.log(err)

        }

      })

  },

  //获取上页信息

    onLoad: function(options) {

        this.setData({

            record\_id: options.record\_id,

            status: app.globalData.status,

        })

        wx.setNavigationBarTitle({

          title: options.name1+'与'+options.name2+'的对话',

        })

    },

  //数据库表onchange绑定函数

  onChange(e) {

    let that = this

    console.log(e)

    if (e.type == "init") {

        let i = e.docs[0].record.length

        i = i - 1

        console.log(i)

      that.initchats(e.docs[0].record)

      if (e.docs.length){

        if(e.docs[0].record.length>0 && e.docs[0].record[i].status != that.data.status){

          chatroomCollection.where({

              \_id: that.data.record\_id,

            })

            .update({

                data:{

                    ['record.'+i+'.isRead']:true

                }

            })

        }

    }

    }

    else {

        let i = that.data.chats.length

        console.log(i)

      const new\_chats = [...that.data.chats]

      if (e.docs.length){

          new\_chats.push(e.docs[0].record[i])

          let j = e.docs[0].record.length - 1

          if(j>=0 && e.docs[0].record[j].status != app.globalData.status){

            chatroomCollection.where({

                \_id: that.data.record\_id,

              })

              .update({

                  data:{

                      ['record.'+j+'.isRead']:true

                  }

              })

          }

      }

      this.setData({

        chats: new\_chats

      })

    }

  },

  initchats(record) {

    this.setData({

      chats: record

    })

    //跳转到页面底部

    this.goBottom()

  },

  //获取输入文本

  getContent(e) {

    this.data.textInputValue = e.detail.value

  },

  sendMsg(){

    let that = this

    var show = false

    if (this.data.chats.length == 0)

      show = true

    //判断上下两条消息的时间差决定是否显示时间

    else {

      if (Date.now() - this.data.chats[this.data.chats.length - 1].sendTimeTS > 120000)

        show = true

    }

    const \_ = db.command

    //消息空白处理

    if (!that.data.textInputValue) {

      wx.showToast({

        title: '不能发送空白信息',

        icon: 'none',

      })

      return

    }

    //消息内容赋值

    const doc = {

      status: app.globalData.status,

      msgText: "text",

      textContent: that.data.textInputValue,

      sendTimeTS: Date.now(),

      showTime: show,

      isRead: false

    }

    //添加数据库表中该记录的records数组，并跳转页面到底部

    chatroomCollection.where({

        \_id: that.data.record\_id

      })

      .update({

        data: {

          record: \_.push(doc)

        }

      })

      .then(res => {

        that.goBottom()

      })

    //消息设空

    that.setData({

      textInputValue: ""

    })

  },

  goBottom() {

    wx.createSelectorQuery().select('#page').boundingClientRect(function (rect) {

      if (rect) {

        // 使页面滚动到底部

        wx.pageScrollTo({

          scrollTop: rect.height + 4

        })

      }

    }).exec()

  },

})

**4.3 功能展示**

详见录屏。