# 4项目详细设计

## 4.1客房管理

### 4.1.1添加客房

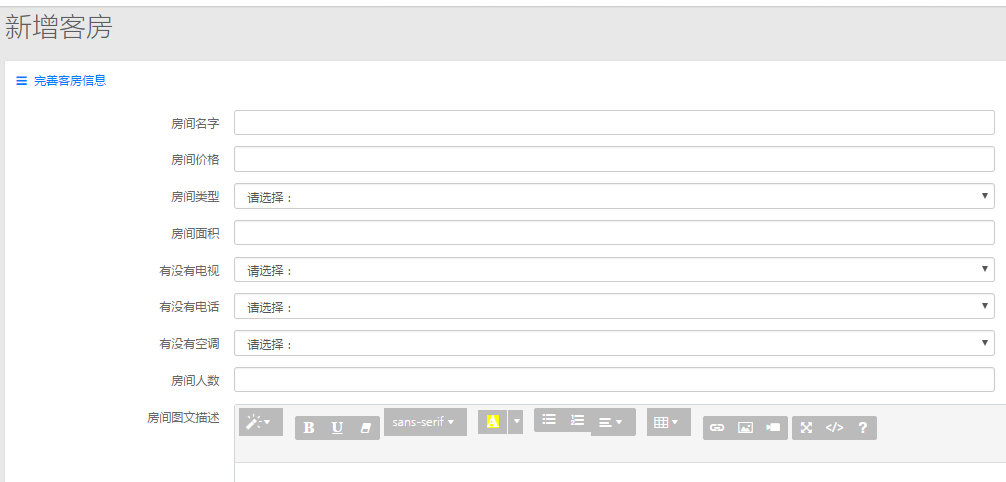
①设计思路

用户点击添加客房之后，会有一个请求发送到我的后台，服务器回传给用户一个表单，让用户填写完整客房信息之后再提交到我们的服务器，这里我使用了富文本编辑器summernote，让用户在网页上随意的对文字和图片进行排版，表单提交到服务器之后，我们调用roomService的insertSelective方法将数据存放进数据库，并回传给用户动作状态标志。

②关键代码

|  |
| --- |
| @RequestMapping("/doadd")  public ModelAndView doadd(RoomWithBLOBs room) {  room.setServiceTime(new Date());  String content = room.getContent();  int firstIndex = content.indexOf("<img src=\"") + 10;  if (firstIndex>=10) {  content = content.substring(firstIndex);  int nextIndex = content.indexOf("\" style=\"width: ");  content = content.substring(0, nextIndex);  room.setPhoto(content);  }  ModelAndView mv = new ModelAndView();    if (roomService.insertSelective(room)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }      mv.setViewName("/sub/room\_add");  return mv;  } |

③界面截图



### 4.1.2管理客房

①设计思路

首先展示给用户所有的客房信息，用户可以查看客房信息，如果有误的话，在表格尾部有修改和删除按钮，点击修改按钮，用户会向服务器发送一个带有房间ID的请求，服务器接收到ID之后，从数据库拿出对应ID的房间信息，放在表单中回传给用户，然后用户修改之后重新提交给服务器，服务器拿到数据之后，覆盖原数据库内容，点击删除按钮，用户会向服务器发送一个带有房间ID的请求，服务器接收到ID之后，从数据库中删除这一条数据。

②关键代码

|  |
| --- |
| @RequestMapping("/doupdate")  public ModelAndView doupdate(RoomWithBLOBs room) {  room.setServiceTime(new Date());    String content = room.getContent();  int firstIndex = content.indexOf("<img src=\"") + 10;  if (firstIndex>=10) {  content = content.substring(firstIndex);  int nextIndex = content.indexOf("\" style=\"width: ");  content = content.substring(0, nextIndex);  room.setPhoto(content);  }      ModelAndView mv = new ModelAndView();    if (roomService.updateByPrimaryKeySelective(room)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }      List<RoomWithBLOBs> allRoom = roomService.all();  mv.addObject("allRoom",allRoom);  mv.setViewName("/sub/room\_all");  return mv;  }  @RequestMapping("/del/{id}")  public ModelAndView del(@PathVariable("id")Integer id) {  ModelAndView mv = new ModelAndView();  if (roomService.deleteByPrimaryKey(id)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }  List<RoomWithBLOBs> allRoom = roomService.all();  mv.addObject("allRoom",allRoom);  mv.setViewName("/sub/room\_all");  return mv;  } |

③界面截图





## 4.2资讯管理

### 4.2.1资讯添加

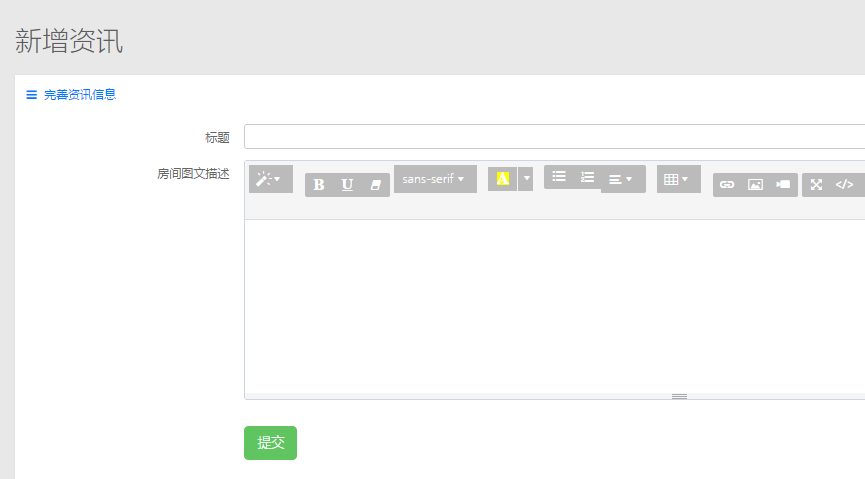
①设计思路

用户点击添加资讯之后，会有一个请求发送到我的后台，服务器回传给用户一个表单，让用户填写完整资讯信息之后再提交到我们的服务器，这里我使用了和客房添加相同的富文本编辑器summernote，表单提交到服务器之后，将数据存放进数据库，并回传给用户动作状态标志。

②关键代码

|  |
| --- |
| @RequestMapping("/add")  public ModelAndView goToAdd() {  ModelAndView mv = new ModelAndView();  mv.setViewName("/sub/new\_add");  return mv;  }    @RequestMapping("/addNew")  public ModelAndView addNew(News news) {  news.setDeleted(0);  news.setUpTime(new Date());  ModelAndView mv = new ModelAndView();  if (newsService.insertSelective(news)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }  mv.setViewName("/sub/new\_add");  return mv;  } |

③界面截图



### 4.2.2资讯管理

①设计思路

首先展示给用户所有的客房信息，用户可以查看客房信息，如果有误的话，在表格尾部有删除按钮，点击删除按钮，用户会向服务器发送一个带有房间ID的请求，服务器接收到ID之后，从数据库中删除这一条数据。

②关键代码

|  |
| --- |
| @RequestMapping("/newsShow")  public ModelAndView newsShow() {  ModelAndView mv = new ModelAndView();  List<News> allNews = newsService.selectAll();  for (News news : allNews) {  news.setContent(stripHtml(news.getContent()));  if (news.getContent().length()>300) {  news.setContent(news.getContent().substring(0,300));  }  }    mv.addObject("allNews",allNews);  mv.setViewName("/sub/new\_all");  return mv;  }        @RequestMapping("/del/{id}")  public ModelAndView newsShow(@PathVariable("id")Integer id) {  ModelAndView mv = new ModelAndView();  if (newsService.deleteByPrimaryKey(id)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }  List<News> allNews = newsService.selectAll();  mv.addObject("allNews",allNews);  mv.setViewName("/sub/new\_all");  return mv;  }    @RequestMapping("/select/{id}")  public ModelAndView select(@PathVariable("id")Integer id) {  ModelAndView mv = new ModelAndView();  News news = newsService.selectByPrimaryKey(id);  mv.addObject("n",news);  mv.setViewName("/sub/new\_select");  return mv;  } |

③界面截图



## 4.3预定管理

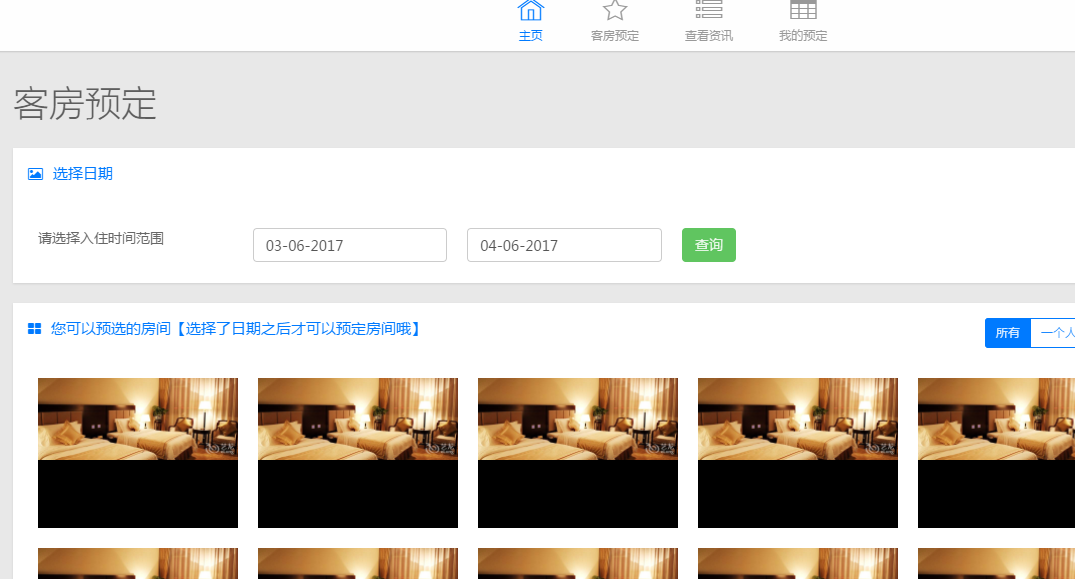
①设计思路

用户首先输入日期范围，提交到服务器，我们在后台根据日期筛选出对应日期的空房间，展示给用户，供用户挑选，用户点击我要预定之后，跳出一个填写信息的窗口，用户将入住人的信息填写完毕之后，即可完成房间的申请，管理员通过审核之后，预定成功，待得客人入住的时候，酒店服务员，点击客人入住的按钮，客人离店就点击客人离店的按钮。服务员还可以强制撤销预定信息。

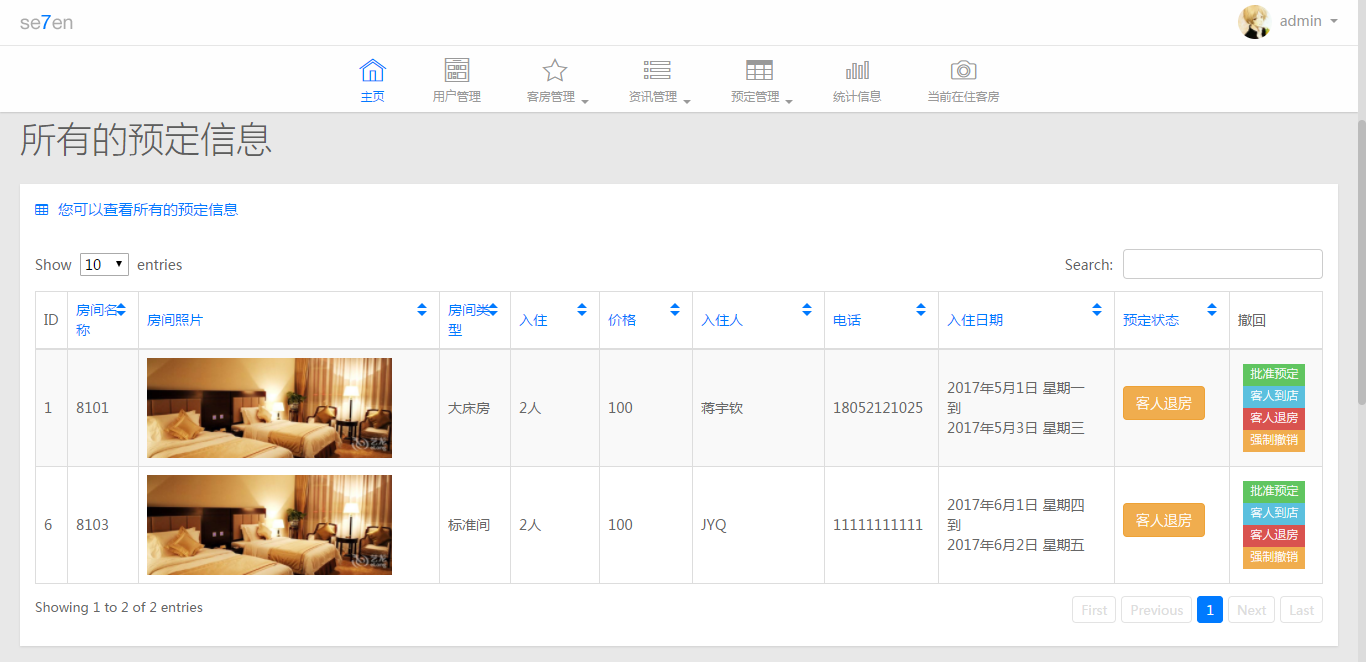
②关键代码

|  |
| --- |
| @RequestMapping("/selectTime")  public ModelAndView selectTime(Order order,HttpSession session) {    session.setAttribute(Constance.timeFlag,order);    List<Order> allOrder = orderService.selectTime(order);    List<RoomWithBLOBs> allRoom = roomService.all();  for (Iterator iterator = allRoom.iterator(); iterator.hasNext();) {  RoomWithBLOBs roomWithBLOBs = (RoomWithBLOBs) iterator.next();  for (Order order2 : allOrder) {  if (roomWithBLOBs.getName().equals(order2.getName())) {  iterator.remove();  }  }  }    ModelAndView mv = new ModelAndView();  mv.addObject("allRoom",allRoom);  mv.addObject("order",order);  mv.setViewName("/sub/room\_show");  return mv;  }      @RequestMapping("/do")  public ModelAndView doit(Order order,HttpSession session) {  session.setAttribute(Constance.timeFlag, null);  order.setOrderTime(new Date());  order.setPs("等待审核");  ModelAndView mv = new ModelAndView();  if (orderService.insertSelective(order)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"预定成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"预定失败！\");");  }    List<RoomWithBLOBs> allRoom = roomService.all();  mv.addObject("allRoom",allRoom);  mv.setViewName("/sub/room\_show");  return mv;  }          @RequestMapping("/my/{id}")  public ModelAndView my(@PathVariable("id")Integer id) {    List<Order> allOrder = orderService.selectByUser(id);  ModelAndView mv = new ModelAndView();  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_my");  return mv;  }      @RequestMapping("/toHotel/{id}")  public ModelAndView toHotel(@PathVariable("id")Integer id) {  ModelAndView mv = new ModelAndView();  Order order = orderService.selectByPrimaryKey(id);    order.setPs("客人入住");  if (orderService.updateByPrimaryKeySelective(order)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }    List<Order> allOrder = orderService.all();  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_all");  return mv;  }      @RequestMapping("/leaveHotel/{id}")  public ModelAndView leaveHotel(@PathVariable("id")Integer id) {  ModelAndView mv = new ModelAndView();  Order order = orderService.selectByPrimaryKey(id);    order.setPs("客人退房");  if (orderService.updateByPrimaryKeySelective(order)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }    List<Order> allOrder = orderService.all();  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_all");  return mv;  }          @RequestMapping("/pass/{id}")  public ModelAndView pass(@PathVariable("id")Integer id) {  ModelAndView mv = new ModelAndView();  Order order = orderService.selectByPrimaryKey(id);  order.setPs("预定成功");  if (orderService.updateByPrimaryKeySelective(order)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }    List<Order> allOrder = orderService.all();  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_all");  return mv;  }          @RequestMapping("/deladmin/{id}")  public ModelAndView deladmin(@PathVariable("id")Integer id) {  ModelAndView mv = new ModelAndView();    if (orderService.deleteByPrimaryKey(id)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"操作失败！\");");  }    List<Order> allOrder = orderService.all();  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_all");  return mv;  }        @RequestMapping("/all")  public ModelAndView all() {    List<Order> allOrder = orderService.all();  ModelAndView mv = new ModelAndView();  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_all");  return mv;  }        @RequestMapping("/allToday")  public ModelAndView allToday() {  List<Order> allOrder = orderService.all();  Calendar cal = Calendar.getInstance();  for (Iterator iterator = allOrder.iterator(); iterator.hasNext();) {  Order order = (Order) iterator.next();  if (cal.getTime().before(order.getToTime())) {    }else if(cal.getTime().after(order.getFromTime()) && cal.getTime().before(order.getToTime())){    }else{  iterator.remove();  }  }      ModelAndView mv = new ModelAndView();  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_all\_t");  return mv;  }        @RequestMapping("/del/{id}")  public ModelAndView del(@PathVariable("id")Integer id,HttpSession session) {  ModelAndView mv = new ModelAndView();  Users user = (Users) session.getAttribute(Constance.userLoginFlag);  if (orderService.deleteByPrimaryKey(id)>0) {  mv.addObject("message", "window.parent.frames.toastrInfo(\"撤销成功！\");");  } else {  mv.addObject("message", "window.parent.frames.toastrInfo(\"撤销失败！\");");  }    List<Order> allOrder = orderService.selectByUser(user.getUserId());  mv.addObject("allOrder",allOrder);  mv.setViewName("/sub/order\_my");  return mv;  } |

③界面截图







## 4.4统计信息

①设计思路

展示酒店近10天的预定情况和入住情况，后台锁定今天的时间，然后逐次往前遍历10天，把相应的信息从数据库取出，回传给用户。

②关键代码

|  |
| --- |
| @RequestMapping("/all")  public ModelAndView all() {  java.text.SimpleDateFormat format = new java.text.SimpleDateFormat("yyyy-MM-dd");  List<Order> allOrder = orderService.all();  String str1 = "";  String str2 = "";        Calendar cal = Calendar.getInstance();  List<Charts> allC = new ArrayList<Charts>();  for (int i = 0; i < 12; i++) {  Charts c = new Charts();  int ii = 0;  int iii = 0;  c.setDate(format.format(cal.getTime()));  for (Order order : allOrder) {  if (format.format(cal.getTime()).equals(format.format(order.getOrderTime()))) {  ii++;  }    if (cal.getTime().equals(order.getFromTime())) {  iii++;  }else if(cal.getTime().after(order.getFromTime()) && cal.getTime().before(order.getToTime())){  iii++;  }  }  str1 += (ii+", ");  str2 += ("{device: \""+c.getDate()+"\", geekbench: "+iii+" }, ");      allC.add(c);  cal.add(Calendar.DAY\_OF\_MONTH, -1);  }    ModelAndView mv = new ModelAndView();  mv.addObject("allOrder",allOrder);  mv.addObject("allC",allC);  mv.addObject("str1",str1.substring(0,str1.length()-2));  mv.addObject("str2",str2.substring(0,str2.length()-2));  mv.setViewName("/sub/charts");  return mv;    } |

③界面截图

