1. Web端轨迹管理系统设计与实现
   1. 系统登录和退出

用户登录界面  
用户在使用系统时，先进入如index.html登录页面，在输入框中输入自己用户名和密码，选项不能为空，为空时浏览器会给出相应提示，然后点击登录。浏览器使用jquery封装的ajax方法来将数据发送给服务器（系统中浏览器和服务器发送和接收数据的方式都是使用该方法来完成的，并且交互的数据格式都是json），UserLoginServlet类将对用户输入的用户名和密码进行判断，然后用Out对象的printJson方法将结果格式化为json字符串，而Out类使用的Json类中的write方法，最终借助jackson.jar库来完成这一操作，最后将结果返回给浏览器，浏览器根据返回结果来判断是否跳转至trackRecord.jsp轨迹管理页面。

在UserLoginServlet类中，将SjyfiUserDao类对象、拼接好的sql查询语句以及包含用户名和密码的列表传递给DBUtil类中的query函数，来从数据库中查询数据，该方法是用来查询一条记录的，包含三个参数，第一个参数是实现了AEntityDao抽象类的子类的对象，第二个参数是和PreparedStatement对象相关的带参数的sql字符串，第三个参数则是对应sql字符串中的参数值列表。AEntityDao类是一个从数据库中返回的ResultSet中获取结果的类，该类有一个抽象方法getEntity，子类必须实现，它规定了如何从ResultSet集合中获取一条记录，另外一个已经实现的方法getEntityList则借助getEntity来获取整个集合。任何从数据库中获取数据的类都是继承了AEntityDao类，SjyfiUserDao类就是AEntityDao类的子类。在DBUtil类里还有个 queryBatch查询方法，通过它来获取多个记录，参数和DBUtil.query一样。DBUtil使用的是jdbc驱动来连接数据库。另外DBUtil包括了三个向数据库插入记录的方法，insert函数插入一条记录、insertAndReturnAutoIncreaseId插入一条记录并返回自增的索引、insertBatch批量插入记录；两个删除记录方法，delete和deleteBatch；以及两个更新方法，update和updateBatch。这几个方法的参数都是sql字符串和对应的sql参数的值两个参数。

PreparedStatement类是继承至Statement类，该对象包含了已编译的sql字符串，可以用来提高执行效率以及提高安全性。

当用户进入系统之后，选择了“退出”选项，由服务器的UserLogOutServlet来接收用户的退出请求，该类将会对该用户进行注销。

登录的流程图：

退出流程图：

* 1. 轨迹管理模块

轨迹管理页面是trackRecord.jsp，该页面分为四个块，按条件搜索轨迹块、显示搜索结果列表块、百度地图（轨迹展示）块以及用户的一些其它操作块。在登录之后，默认会话的超时时间是20分钟，如果用户在20分钟内为进行任何操作，系统将会自动掉线，之后用户再次操作时，系统会给出超时提示，并返回登陆页。轨迹管理界面如下图所示：

* + 1. 轨迹上传

文件上传使用的是Plupload UI插件，上传界面如下图所示。

在进行文件上传前，用户需要先在移动终端的eScienceApp\_DataCollector/routeRecord/ routeZip目录下，将kmz文件拷贝至电脑上，之后再登陆到系统。进入系统后，点击轨迹管理页面右上角的“文件上传”按钮，弹出上传界面；选择“增加文件”项，弹出选择文件框，用户选择自己要上传的kmz文件，选择成功后，点击“打开”；浏览器端先检测用户选择的文件类型是否为kmz文件；然后再由服务器的CheckFileExistServlet类确认用户所选择的文件是否在服务器端已存在；当用户所选文件不符合要求时，将会给出相关提示，只有符合条件的文件才能进行上传，上传的文件由UploadFileServlet类进行处理，该类借助commons-fileupload.jar库来接收文件；当文件上传成功之后，调用JFile类的addUploadFilePath方法将上传的文件路径添加到JFile类中，JFIle类将会去读取该目录将该上传文件解压至指定的目录，然后调用TrackDetailFileParse类来解析TrackDetail.xml文件中的数据，以及调用PlaceMarkFileParse类解析RouteRecord.kml文件，并将解析的信息插入到数据库中的相关表中。

在浏览器进行文件上传的时候，会判断浏览器是否支持分块上传，如果支持，将会按照指定的大小对文件进行分块上传，并且如果文件在上传过程中失败，将会自动尝试重传3次，3次之后仍上传失败，将给出文件上传失败信息。

当用户上传的文件较大时，解压文件以及从文件中提取数据需要一定的时间，为了不影响用户的体验，JFile类是一个继承Thread线程类，在JFile类的addUploadFilePath方法中，会检测JFile线程是否已经在解压文件，如果不在的话，将会开启JFile线程来解压文件。

在解析文件时，使用的是SAX（Simple API for XML）接口来解析文件，由于SAX是基于事件驱动的解析器，内存消耗少，以此来提高系统的效率。DefaultHandler类是SAX的提供的一个默认基类，该类中有五个基本的方法，startDocument方法，当开始解析文档是会执行方法；endDocument解析文档结束时执行；startElement方法，当开始解析节点时执行；endElement结束解析节点时会触发的事件；characters保持节点内容方法，该方法是在startElement后以及endElement前，在对应的startElement和endElement之间该方法可能会触发多次。要解析文件时只要继承DefaultHandler类，然后实现这五个方法便能够提取相关数据。DefaultSaxParse就是一个继承至DefaultHandler的类，该类中包含了一个BaseFileParse抽象类的一个实例，在BaseFileParse声明了DefaultHandler类中的那五个基本方法，DefaultSaxParse类则是调用BaseFileParse对象中的方法来完成对数据的解析。当在解析数据时，只需要继承BaseFileParse基类，然后实现它，再将实现了BaseFileParse基类方法的子类对象传递给DefaultSaxParse类，便可以解析数据。借助JSAXParser类来完成这一传递过程，该类提供一个parse方法，含有两个参数，一个是要解析文件的路径和一个实现BaseFileParse基类的子类的对象。TrackDetailFileParse和PlaceMarkFileParse类都是继承BaseFileParse基类。

上传文件的流程图：

* + 1. 轨迹搜索

轨迹搜索包括按照时间、地点、记录人、区域范围（即经纬度的范围）等多种方式进行搜索，也可以组合条件进行搜索。

在进行轨迹搜索时，用户先在界面上选择自己的搜索方式，然后输入相关的关键字，点击“搜索”按钮进行查询，浏览器将用户输入的查询关键字拼接成事先预订好的json字符串格式发送给QueryRecordServlet类，该类将借助ConditionEntity类的getSql方法来得到sql查询语句和getSqlValues方法来得到sql语句相应参数的值，以及TTracksDao类对象（TTracksDao是一个继承至AEntityDao的类）这三个参数传递给DBUtil的queryMulti方法来执行查询并将结果返回给浏览器。浏览器将结果显示在页面的搜索结果列表中。在未得到查询结果之前，禁止用户再次发送查询请求。

由于搜索得到的信息较少，可以通过点击搜索列表中每条记录的名称来查询具体信息，通过ViewSingleRouteRecordInfoServlet类来获取，该类借助TTracksDao类以及DBUtil类的query方法来查询数据库。

轨迹搜索流程图：

* + 1. 轨迹展示

用户在搜索列表的“显示”列选择要展示的轨迹，浏览器会先判断该轨迹是否已经加载过，如果是第一次加载则会发送数据到服务器。然后由RouteRecordMapInfoServlet类来接收用户选择的记录id，在该类中，同样使用TTracksDao类对象、拼接的sql字符串和sql字符串对应的参数值列表作为参数来调用DBUtil类的query方法来查询该记录所在文件的目录。然后使用PlaceMarkFileParse类按照该目录下的RouteRecord.kml文件的数据格式来解析，将轨迹信息提取出来，然后以json格式返回给浏览器，在浏览器端的jmap脚本，将对返回的json进行解析，该脚本中有个loadData方法来加载数据，生成轨迹线段以及关键点标注并在百度地图上显示，用户可以点击关键点来查看具体信息，包括图片、视频和音频，对于视频和音频也可以下载，以免视频和音频不能播放时，用户可以下载到本地浏览。当查询失败时，搜索列表中的该条记录颜色会变为红色。在查询期间，禁止用户查询其它记录的轨迹。

轨迹展示流程图：

* + 1. 轨迹合并

系统根据用户的角色，来提供轨迹合并并直接导入数据库以及轨迹合并功能。对于轨迹合并，任何用户都可以使用该功能；但是只有管理员用户，才提供轨迹合并并直接导入数据库。在页面的右上角的“导出轨迹”按钮就是用来合并轨迹，“保存轨迹”按钮则是导入数据库。

在轨迹合并中，用户必须在搜索列表中选择以条轨迹记录以上才能使用合并功能。由服务器ExportRecordServlet类来处理用户的这一请求，该类通过DBUtil类的queryMulti方法来查询请求记录的文件目录。如果用户选择一条记录时，就根据目录直接返回该条记录的kmz文件；多条记录时，把合并完成后的文件所在目录和要合并的目录列表传递给FileMerge类的work方法来合并文件，在调用JZipFile的work方法在压缩文件。在合并文件过程中，需要按照格式提取各个文件下的RouteRecord.kml以及提取TrackDetail.xml文件的信息来生成新的对应的文件，其它文件则依次拷贝至合并目录。在提取文件信息中，FileMerge类调用RouteRecordFileParseAndMerge类来解析合并RouteRecord.kml文件和调用TrackDetailFileParseAndMerge类来提取并合并TrackDetail.xml文件，这两个类都是继承至BaseFileParse基类。文件合并完成后，就进行文件压缩。压缩后将告知浏览器文件已经合并完成，即将下载，这时浏览器会自动发送信息给DownloadFileServlet类，请求下载文件，该类调用DownloadFile类的work方法来将压缩后的文件返回给用户。

管理员的轨迹导入数据库功能则是轨迹合并的升级版，当文件合并以及压缩成功之后就会将TrackDetail.xml信息提取然后保存至数据库。这个功能是通过SaveRecordServlet类来完成。当选择“保存轨迹”这一选项时，用户必须选择两条以上记录才有效。

在轨迹合并和压缩的过程中，如果文件较多或者较大时，将会比较耗时，为了提高系统性能和反应时间。在合并文件过程中，对于RouteRecord.kml文件的解析，按照该文件中的“Plackmark”标记来将文件分成一个个只含“Plackmark”标记的小文件，最后再将小文件合并成新的RouteRecord.kml文件；而在TrackDetail.xml文件中信息较少，则是直接提取信息保存到内存中，最后写入到新的TrackDetail.xml文件中。压缩文件使用的是apache的ant.jar库来完成的，相比自己写的压缩功能，该库有着明显的压缩优势。