AITA

软件工程专业综合项目概要设计

1652714 孙浩然

1652763 陈泽徽

1652751 梁钧清

1553545 王嵩豪

1652708 周泽林

1652698 康晓博

# 目录

[AITA 1](#_Toc534129803)

[目录 2](#_Toc534129804)

[引言 3](#_Toc534129805)

[概要设计依据 3](#_Toc534129806)

[参考资料 3](#_Toc534129807)

[假定和约束 4](#_Toc534129808)

[概要设计 4](#_Toc534129809)

[系统总体架构设计 4](#_Toc534129810)

[系统软件结构设计 6](#_Toc534129811)

[结构语境图 6](#_Toc534129812)

[系统业务分析类图 7](#_Toc534129813)

[课程管理子系统 7](#_Toc534129814)

[学习结果管理子系统 8](#_Toc534129815)

[用户管理子系统 8](#_Toc534129816)

[系统软件架构图 9](#_Toc534129817)

[接口设计 9](#_Toc534129818)

[内部接口 9](#_Toc534129819)

[外部接口 12](#_Toc534129820)

[界面设计 13](#_Toc534129821)

[数据库设计 16](#_Toc534129822)

[系统出错处理设计 18](#_Toc534129823)

[出错信息 18](#_Toc534129824)

[补救措施 19](#_Toc534129825)

[系统维护设计 19](#_Toc534129826)

# 引言

## 概要设计依据

* AITA——需求规约文档
* AITA——需求分析规约文档

## 参考资料

[1] 质量管理体系国家标准理解与实施(2008版)

[2]Jeon, J., Park, J. C., Jo, Y. J., Nam, C. M., Bae, K. H., & Hwang, Y., et al. (2016). A Real-time Facial Expression Recognizer using Deep Neural Network.International Conference on Ubiquitous Information Management and Communication(pp.1-4). ACM.

[3]Kim, B. K., Roh, J., Dong, S. Y., & Lee, S. Y. (2016). Hierarchical committee of deep convolutional neural networks for robust facial expression recognition.Journal on Multimodal User Interfaces,10(2), 1-17.

[4]Chang, T., Wen, G., Hu, Y., & Ma, J. J. (2018). Facial expression recognition based on complexity perception classification algorithm.

[5]Li, D., Wen, G., Hou, Z., Huan, E., Hu, Y., & Li, H. (2018). Rtcrelief-f: an effective clustering and ordering-based ensemble pruning algorithm for facial expression recognition.Knowledge & Information Systems, 1-32.

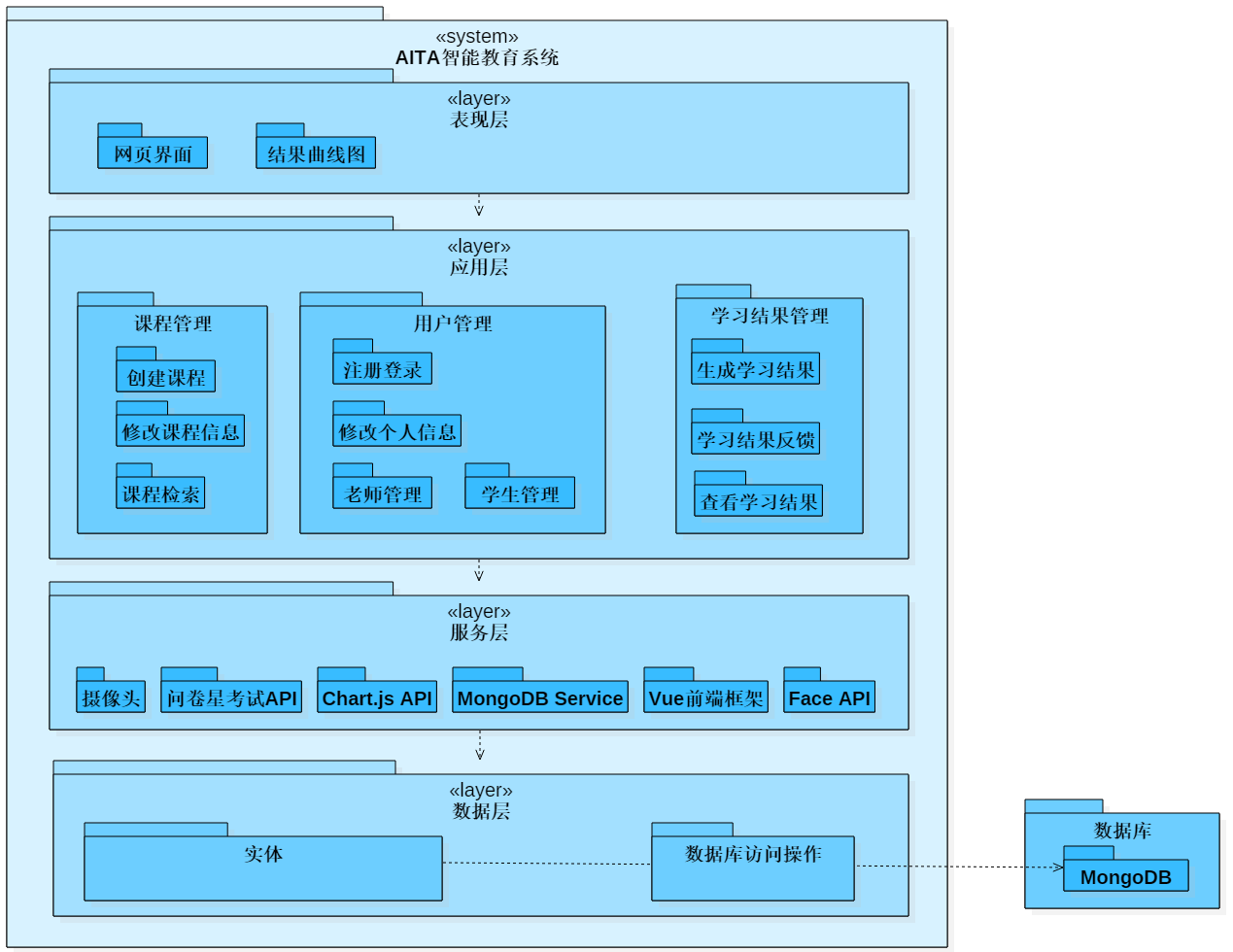
## 假定和约束

开发期限：2019年6月20日

设备条件：具有摄像头的PC机

# 概要设计

## 系统总体架构设计



本架构以**MVC框架**为指导，将系统分为表现层、应用层、服务层、数据层四个层次。

**表现层**为系统的webGUI表面和图形化显示部分，接受用户操作等信息，通过HTTP协议与应用层进行通信。

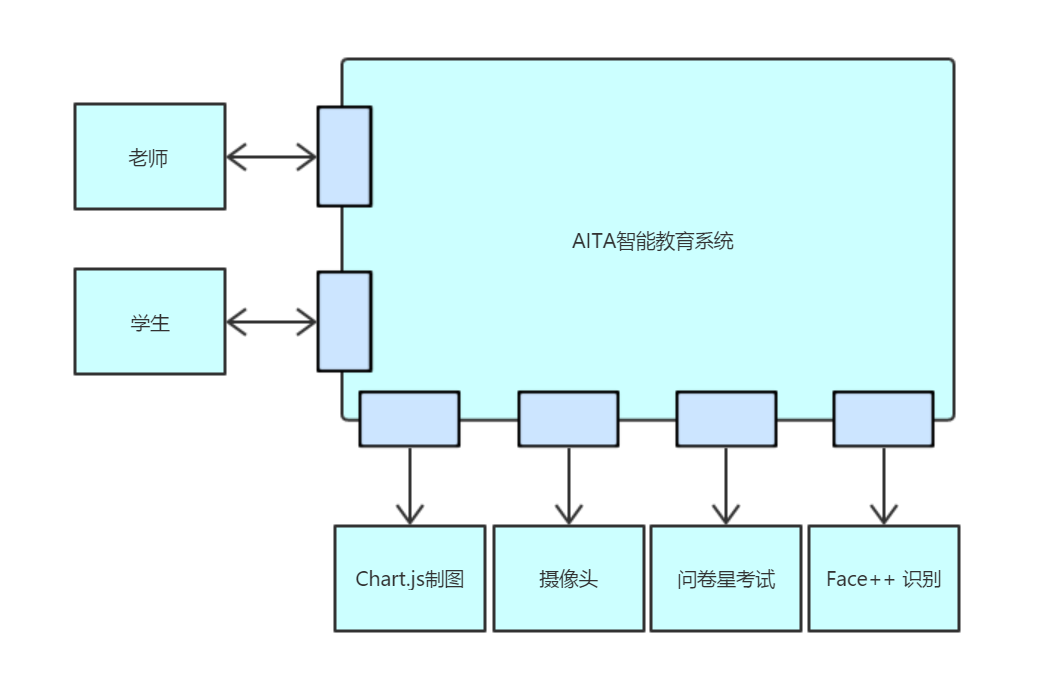
**应用层**为系统主要业务逻辑，处理表现层接收的前端请求，同时控制前端的显示内容，对前后端进行连接。系统主要业务分为三个子模块：课程管理、用户管理、学习情况管理。

**服务层**为应用层提供支持，包括前后端框架和系统使用到的第三方API的管理。

**数据层**作为MVC中的Model部分，提供实体类对象的持久化存储和读取方法。数据库访问操作中封装了对MongoDB数据库的链接、查询等基本操作，其它层在需要与数据库进行交互式只调用数据库访问。

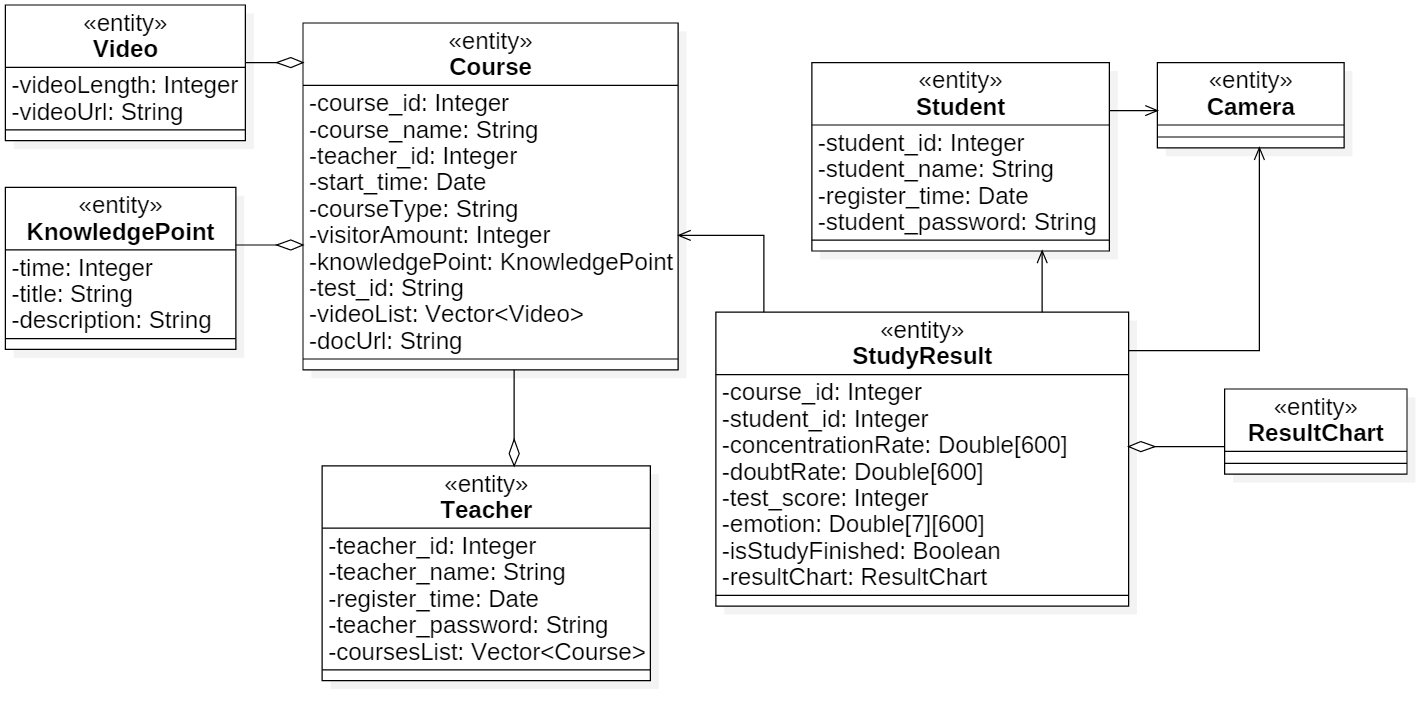
## 系统软件结构设计

### 结构语境图



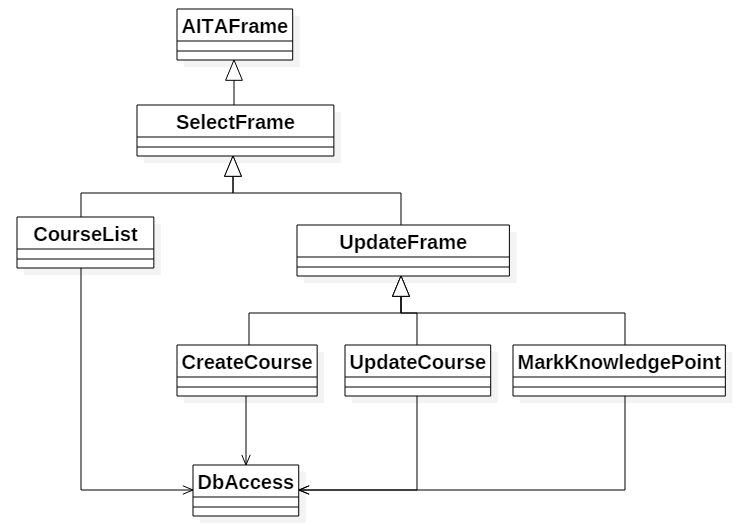
AITA智能教育系统是一个相对独立的平台，与老师和学生用户进行交互，但几乎不与其他系统构成 peer 或 superordinate 关系；系统在使用过程中需要调用外部的系统来进行识别校准、绘图、摄像、生成试卷功能。

### 系统业务分析类图

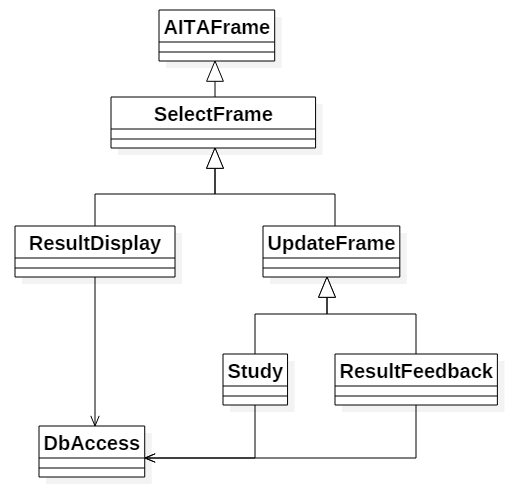


根据系统业务可以将整个系统分为课程管理、学习结果管理、用户管理三个子系统。

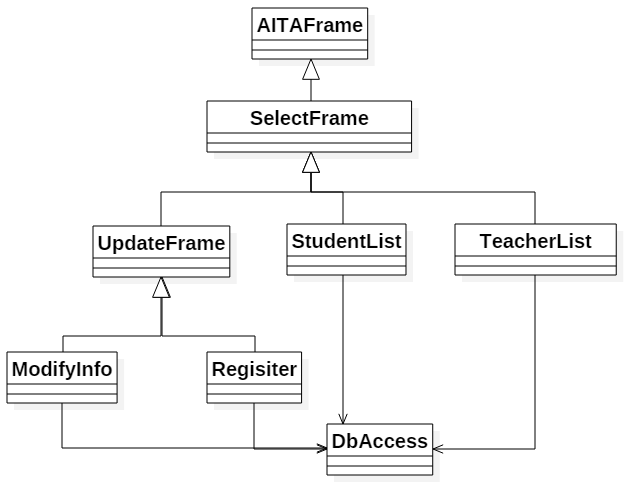
### 课程管理子系统



### 学习结果管理子系统

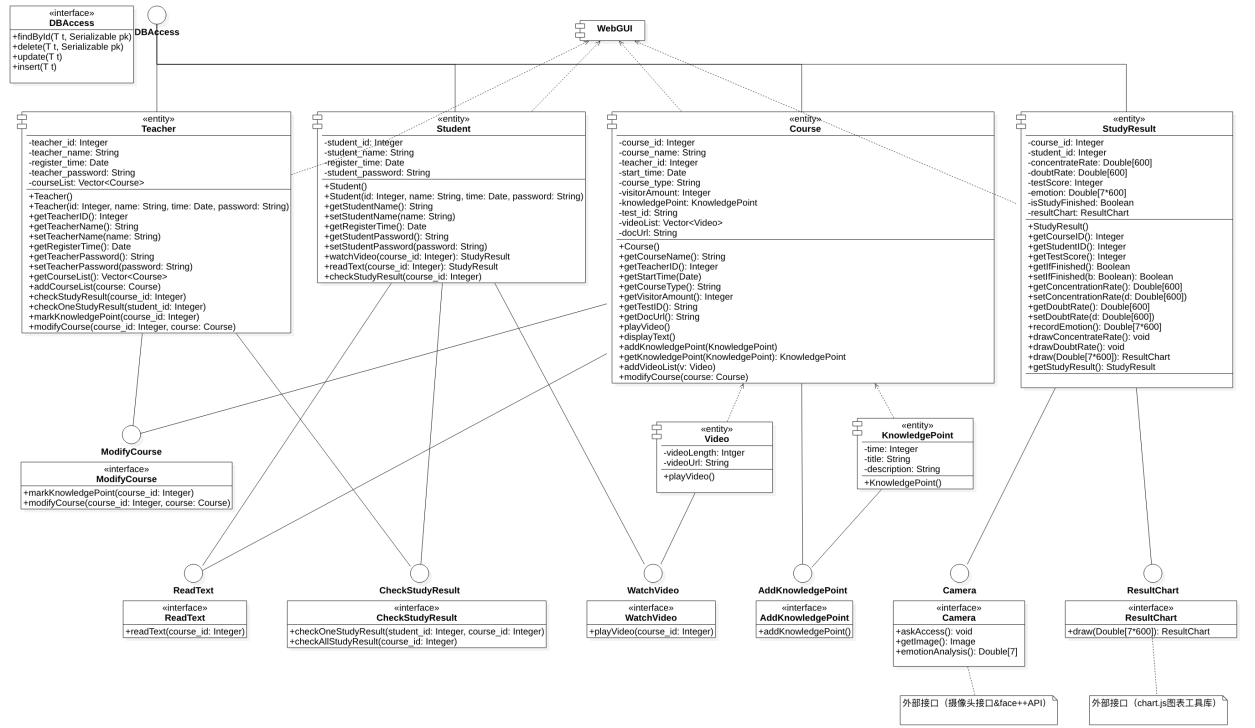


### 用户管理子系统



通过problem frame继承的形式针对每个子系统初步构建设计类图。

### 系统软件架构图



将系统业务设计类图初步加入接口类并构成组件，组件之间的层次调用是通过接口实现的。主要业务模块的操作对象抽象为组件，对于需要调用其他组件实现的功能定义接口，通过接口调用实现特定业务功能。

## 接口设计

### 内部接口

实体类从数据库表格映射得到，其属性值为私有类，可以通过对应的 getter和setter进行访问和修改。同时还提供一部分接口完成业务逻辑。

* 《interface》ModifyCourse

ModifyCourse 接口用来实现 Teacher 类与 Course 类之间方法的调用：

Teacher 类的markKnowledgePoint(course\_id:Integer) 方法通过接口来调用 Course 类的 addKnowledgePoint(KnowledgePoint) 方法 ，完成标记知识点功能。

Teacher 类的 modifyCourse(course\_id:Integer,course:Course) 方法通过接口（modifyCourse(course\_id:Integer,course:Course) ）来调用 Course 类的 modifyCourse(course:Course) 方法 ，完成改进课程功能。

* 《interface》ReadText

ReadText 接口用来实现 Student 类与 Course 类之间方法的调用：Student 类的 readText(course\_id:Integer) 方法通过接口来调用 Course 类的 displayText() 方法 。

* 《interface》CheckStudyResult

CheckStudyResult 接口用来实现 Student 类、 Teacher 类与 StudyResult 类之间方法的调用：Student 类的 checkStudyResult(course\_id:Integer) 方法通过接口（ checkOneStudyResult(student\_id:Integer,course\_id:Integer)）来调用 StudyResult 类的 getStudyResult() 方法 ，得到特定学生的学习结果。

Teacher 类的 checkOneStudyResult(student\_id:Integer)方法通过接口（ checkOneStudyResult(student\_id:Integer,course\_id:Integer) ）来调用 StudyResult 类的 getStudyResult() 方法 ，得到特定学生的学习结果。

Teacher 类的 checkStudyResult(course\_id:Integer) 方法通过接口（ checkAllStudyResult(course\_id:Integer) ）来调用 StudyResult 类的 getStudyResult() 方法 ，得到选择此门课程的所有学生的学习结果。

* 《interface》WatchVideo

WatchVideo 接口用来实现 Student 类与 Course 类之间方法的调用：Student 类的watchVideo(course\_id:Integer) 方法通过接口来调用 Course 类的 playVideo() 方法 。

* 《interface》AddKnowledgePoint

AddKnowledgePoint 接口用来实现 Course 类与 KnowledgePoint 类之间方法的调用：Course 类的addKnowledgePoint(KnowledgePoint) 方法通过接口 中的方法 addKnowledgePoint() 来调用 KnowledgePoint 类实现标记知识点。

对数据库的访问、操作，通过 DBAcess 提供的接口来实现，DBAccess为抽象接口，具体数据访问类要重写该接口以方便调用。

* DBAccess

insert(T t) 保存/新建某个对象到数据库；

delete(T t, Serializable pk) 删除数据库的某个对象；

findById(T t, Serializable pk) 由主码查找某些数据；

update(T t) 更新某条记录。

### 外部接口

* 《interface》Camera

Camera 是外部摄像头接口，askAccess() 用来获取用户的摄像头访问权限，getImage(): Image用来捕捉学习时的图像（1s/次），emotionAnalysis() 用来调用 face++API 实现表情分析，返回七中表情对应的概率 Double[7]。

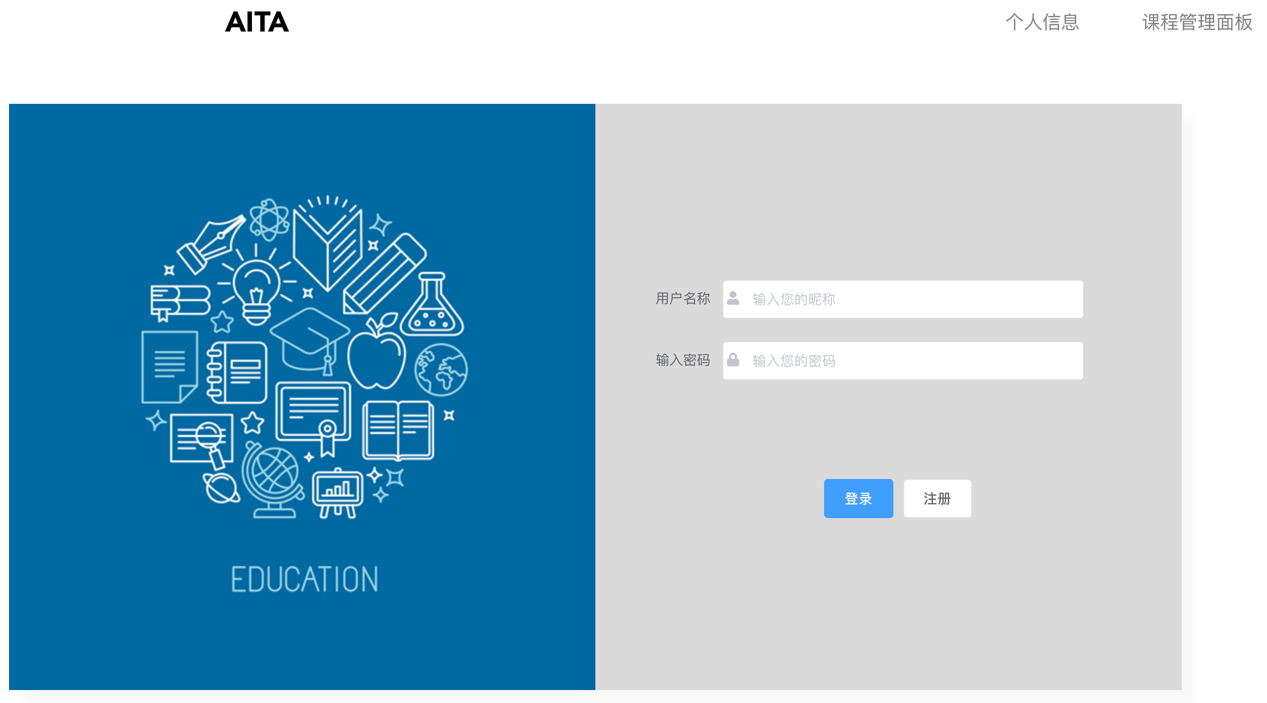
Course类的 recordEmotion() 方法实现此接口来获得表情信息，以供根据表情数据进行学习情况分析。

* 《interface》ResultChart

ResultChart 是绘制图表的外部 Chart.js 接口，Course类的 draw(Double[7\*600]) 方法实现此接口来绘制学习情况图表。

## 界面设计

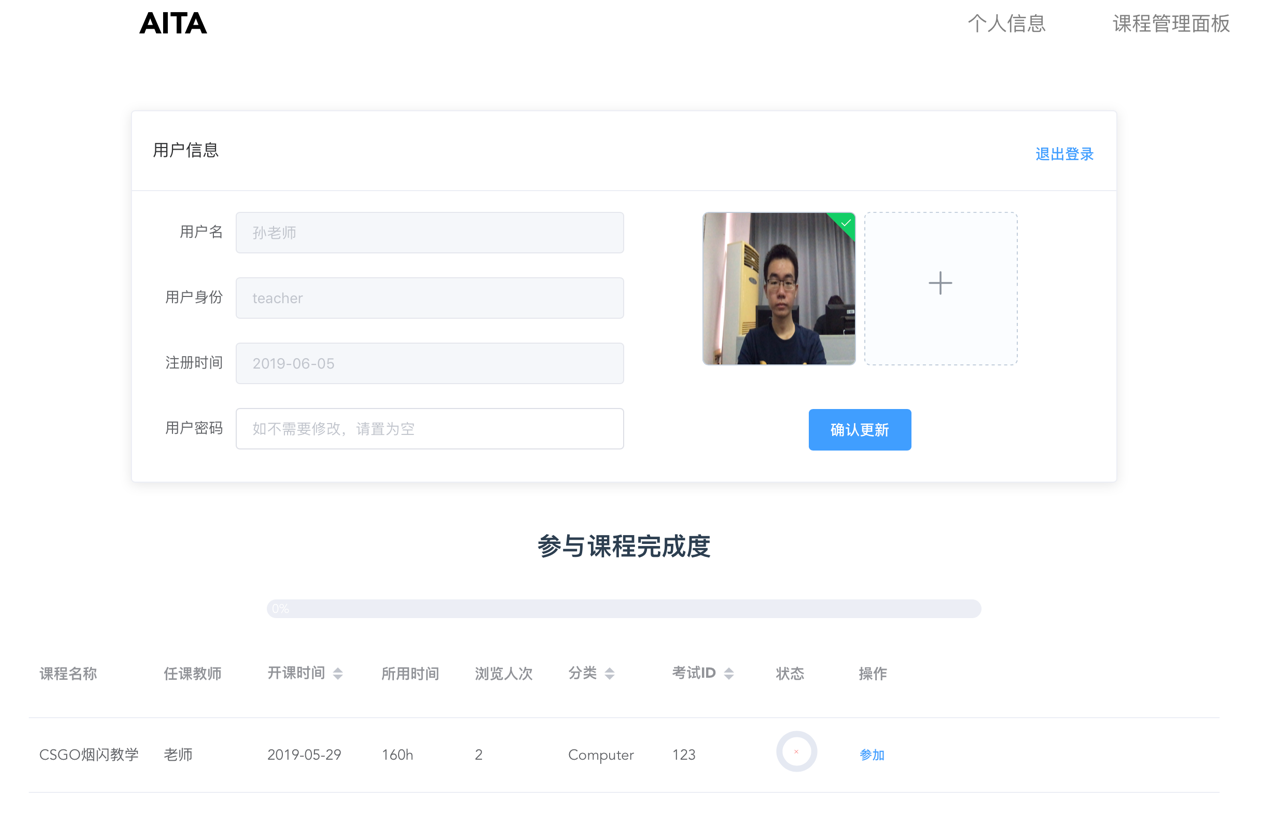
登录界面：



注册界面：



用户管理：

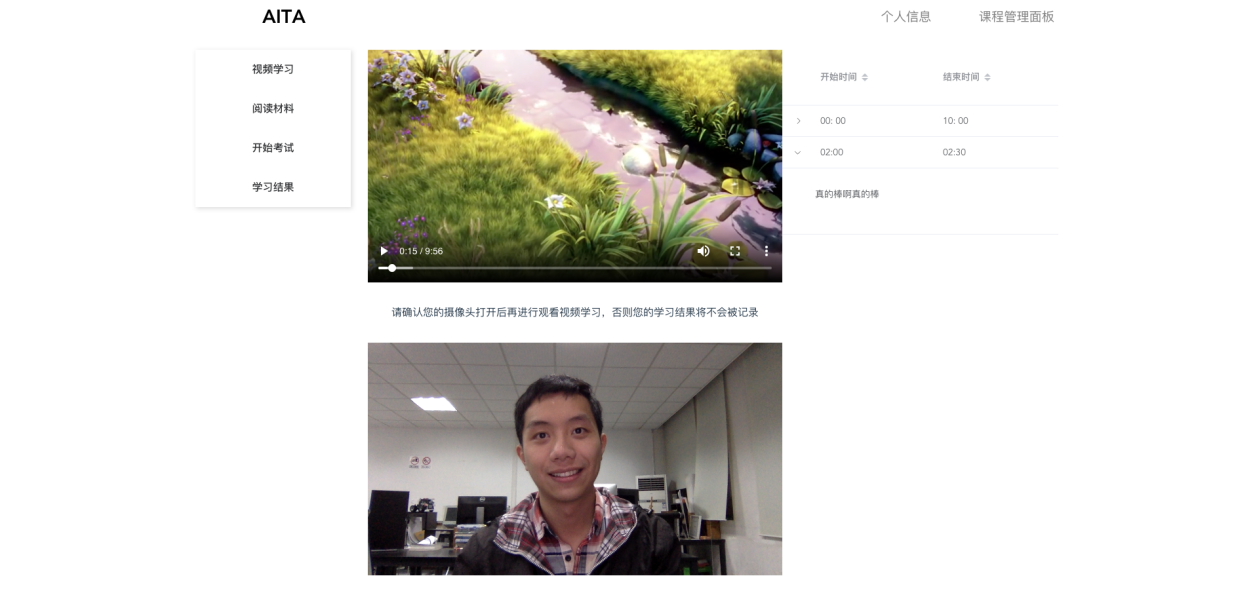


课程管理：

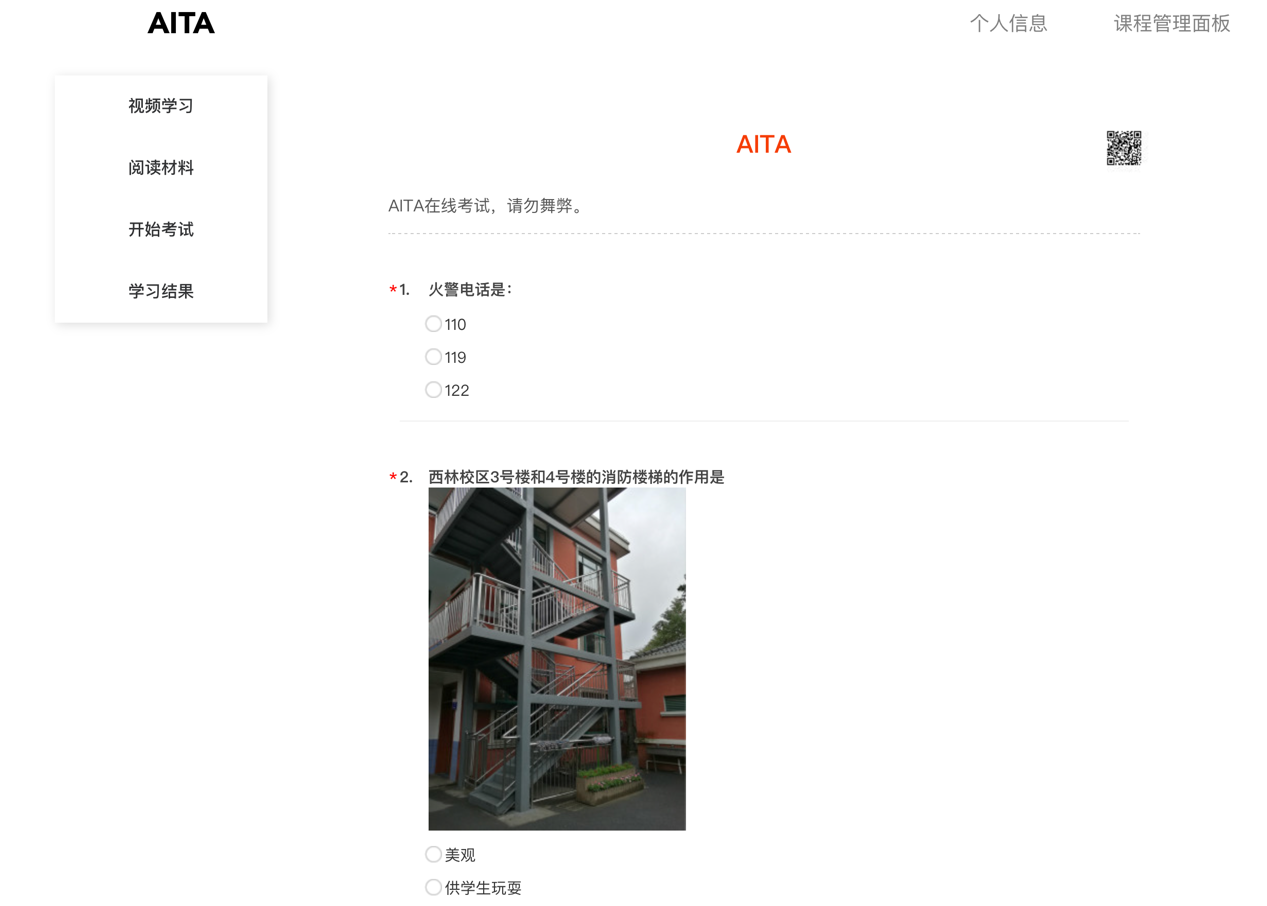


上传课程：



观看视频页面：

考试页面：



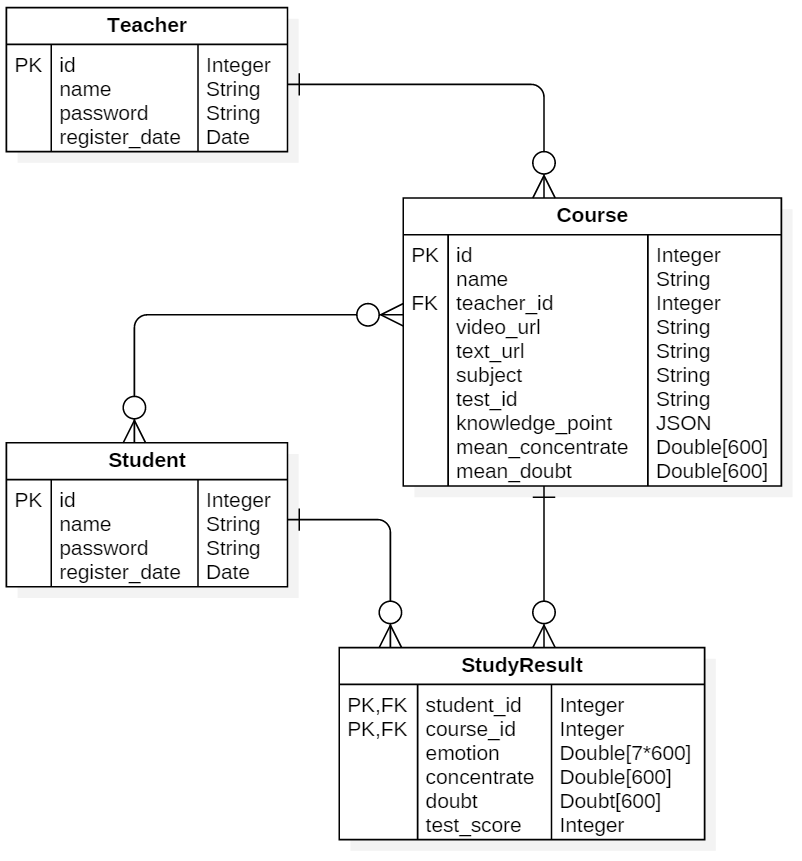
老师查看课程学习情况：



学生查看学习结果页面：



## 数据库设计



表Teacher：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **主键/外键** | **属性** | **数据类型** | **非空** | **默认值** | **备注** |
| 主键 | id | Integer | Not Null |  | 老师编号 |
|  | name | String | Not Null |  | 老师账户名 |
|  | password | String | Not Null |  |  |
|  | register\_date | Date | Not Null |  | 注册日期 |

表Student：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **主键/外键** | **属性** | **数据类型** | **非空** | **默认值** | **备注** |
| 主键 | id | Integer | Not Null |  | 学生编号 |
|  | name | String | Not Null |  | 学生账户名 |
|  | password | String | Not Null |  |  |
|  | register\_date | Date | Not Null |  | 注册日期 |

表StudyResult:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **主键/外键** | **属性** | **数据类型** | **非空** | **默认值** | **备注** |
| 主键/外键 | student\_id | Integer | Not Null |  | 学生编号 |
| 主键/外键 | course\_id | Integer | Not Null |  | 课程标号 |
|  | emotion | Double[7][600] |  | null | 表情识别结果，在十分钟的视频中每秒解析并存储一个JSON数据 |
|  | concetrate\_rate | Double[600] |  | null | 每秒根据表情识别结果生成一个对应的专注度值 |
|  | doubt\_rate | Double[600] |  | null | 每秒根据表情识别结果生成一个对应的疑惑度值 |
|  | test\_score | Integer |  | null | 学生考试分数 |

表Course：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **主键/外键** | **属性** | **数据类型** | **非空** | **默认值** | **备注** |
| 主键 | id | Integer | Not Null |  | 课程编号 |
|  | name | String |  | 未命名 | 课程名 |
| 外键 | teacher\_id | Integer | Not Null |  | 创建课程的老师 |
|  | video\_url | String |  | null | 课程视频路径 |
|  | text\_url | String |  | null | 课程文档路径 |
|  | subject | String |  | null | 课程所属科目 |
|  | test\_id | Integer |  | null | 课程测验编号，来自问卷星 |
|  | knowledge\_point | JSON |  | null | 老师在视频中标记的知识点  {  time:Integer  topic:String  description:String  } |
|  | mean\_concentrate | Double[600] |  | null | 所有上课学生平均专注度 |
|  | mean\_doubt | Double[600] |  | null | 所有上课学生平均疑惑度 |

## 系统出错处理设计

### 出错信息

|  |  |  |
| --- | --- | --- |
| 出现错误 | 系统输出信息 | 错误处理方式 |
| 注册时用户信息不符合规定 | 输出相应的错误信息 | 禁用注册按钮 |
| 登录时用户名和密码不匹配 | 提示“用户名或密码错误” | 回滚之登录页面 |
| 注册用户名重名 | 后端返回false | 禁用注册按钮 |
| 未登录时的参加课程 | 返回JSON字符串“haven't  log in.” |  |
| 查询课程无此课程 | 返回JSON字符串“No available words. ” |  |
| 查看学习情况失败 | 返回JSON字符串“抱歉，我们无法对您的学习情况进行有效汇报. ” |  |

### 补救措施

* 注册时用户信息不符合规定

系统将会在该输入右侧提示错误原因，诸如“用户名只能包括数字、英文、下划线”，“两次 输入密码不一致”等提示性语句。 若不能准确填写所有注册信息，注册按钮处于被禁用状态

* 注册用户名重名

后端会返回false，注册按钮处于被禁用状态

* 登录时用户名、密码不匹配

系统将会提示“用户名或密码填写错误”，并回滚至登录界面，要求再次输入。

### 系统维护设计

* 注册信息格式的判断，写于前端的Javascript中，进行检测。
* 注册时，判断用户名是否重名，写于webAPP中，通过发送Post请求，访问DBAccess，从而对数据库中已存信息进行查询。
* 登录时判断用户名、密码是否匹配，写于webAPP中，通过发送Post请求，访问DBAccess，从而对数据库中已存信息进行查询。
* 根据学生对于学习情况的反馈，优化推导学习情况的算法。
* 用户更新图片时，增加对用户更新前后的两个两张图片人脸面部识别对比