SOFTWARE DESIGN DOCUMENT for iSport

Release 1.0

Prepared by

1650940 Jiang Xiaohu
1650932 Xu Jingnan
1651058 Wang Yicheng

School of Software Engineering
Tongji University

December 26, 2019

Table Of Contents

Revision History

Name	Date	Reason For Changes	Version
Jiang Xiaohu	2019.11.12	Finish Introduction Part	v1.0
Wang Yicheng	2019.11.14	Finish Overview Part	v1.1
Xu Jingnan	2019.11.16	Finish External Requirements Part	v1.2
Xu Jingnan, Jiang Xiaohu, Wang Yicheng	2019.11.18	Finish Sequence Diagrams for Function Modeling Part	v1.3
Xu Jingnan	2019.11.20	Finish Function Modeling Part	v1.4
Jiang Xiaohu	2019.11.21	Finish Data Modeling Part	v1.5
Wang Yicheng	2019.11.22	Finish Behavior Requirements Part	v1.6
Xu Jingnan	2019.11.23	Finish Nonfunctional Requirements Part	v1.7
Jiang Xiaohu	2019.11.25	Finish Data Dictionary Requirements Part	v1.8

1 Introduction

1.1 Purpose

This design document describes the overall structure of the Class Collaboration Application (CCA) by outlining significant aspects of the system's architecture.

1.2 Scope

This application will be used for students at Case Western Reserve University to collaborate and discuss specific courses they are in with other students. Some of the key features of this application are that students will be able to chat with other students in the course, upload class notes or other miscellaneous documents to discuss, and ask questions. Users will also be able to download any documents uploaded for a course that they are enrolled in and can personalize their own account on the CCA.

1.3 Acronyms, Abbreviations and Definitions

CCA (Class Collaboration Application) - Acronym of the name of the application.
PaaS (Platform as a Service) - Acronym of a cloud computing platform.
IaaS (Infrastructure as a Service) - Acronym of a cloud computing infrastructure.

VM (Virtual Machine) - Acronym of virtual machine.
SSO (Single Sign-On) - Acronym of a tool for access control across several independent software systems.

1.4 Overview

This document is an overview of the software architecture of the Class Collaboration Application in high detail. We start by providing all the principal classes that the application is built on as well as their responsibilities to the success of the application. Next, we provide diagrams to show the hierarchy of our classes and the architectural design. Finally, we develop a general API of the major class methods used to build the functionality of our application. There are also mockups of our UI design included in our design document.

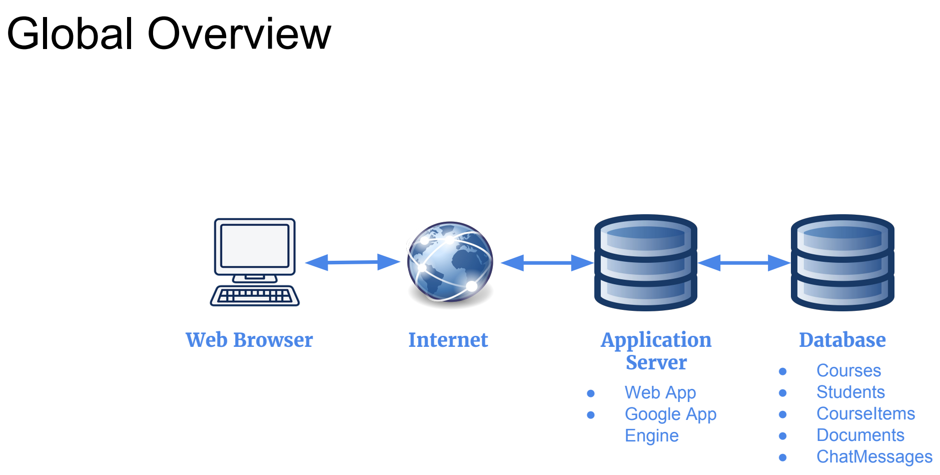
1.5 Reference Material

Book Inventory System System Design Document. December 8 2010.
ZZZ Company. *ZZZ Software Architecture.* October 2 2015.

2 System Overview

3 System Architecture

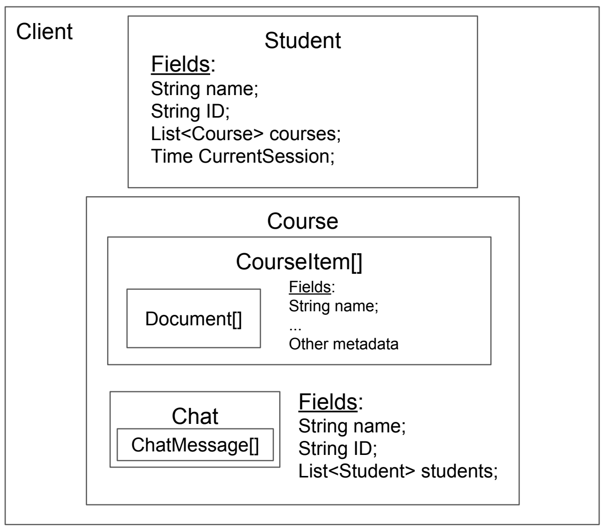
3.1 Architectural Design



3.2 Description of Achitecture Goals

The Class Collaboration Application will be hosted by the Google App Engine and users will access the app via a web browser. The app will have access to a database containing lists of all the Courses created for the app, corresponding CourseItems, Documents, and Chatmessages, as well as the Students who have created accounts.

Class Structures

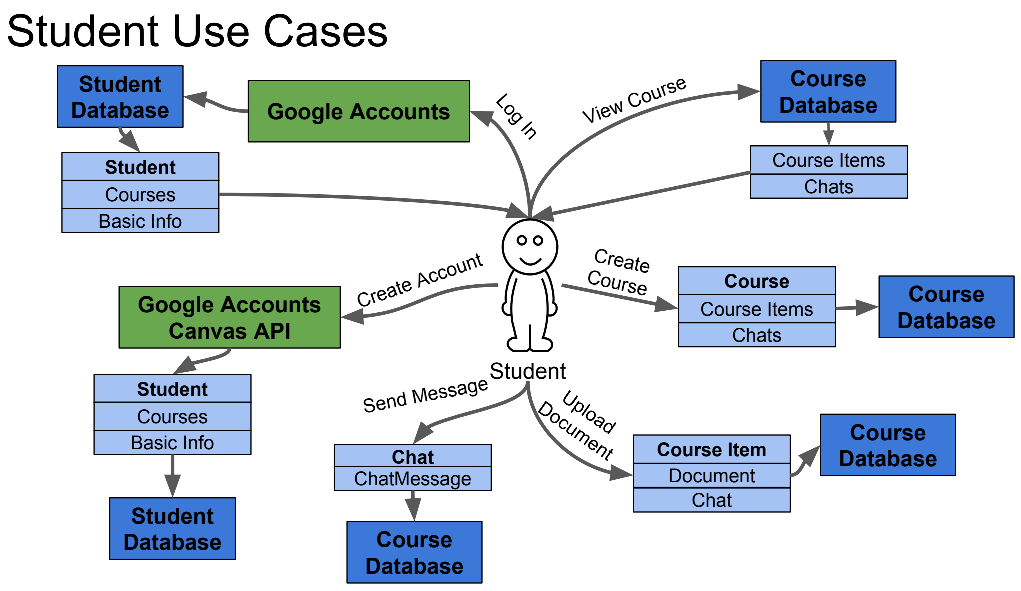


The database will save the information for an indefinite amount of time. Our class structures and main use cases by users are shown below.

3.3 Design Rationale

We have decided to use Google App Engine because it is a reliable platform to scale and build web applications. Many web applications are maintained either through IaaS or PaaS. We decided not to use Iaas because, although we can have root access to a VM, we would have to be responsible for managing the resources on the machine including, memory and CPU usage. Since Google App Engine is a PaaS, it manages all of our computational resources for us, so our only responsibility is maintaining the application while Google App Engine would take care of the infrastructure, security and scalability of the Class Collaboration Application.

For each user, we decided to integrate SSO into our application. Using SSO as a way for users to sign into the application through their Case credentials boosts our security capabilities as well as mitigates the risk of 3rd party applications accessing sensitive information about the user. This also improves user experience since



the user does not have to create and keep track of another username and password.

4 Principal Components

4.1 Student

The student class contains all user information including student ID, the name of the student, the courses they are enrolled in, their username and the last time they logged in. A new student is created by checking whether a specific user has previously logged into the Class Collaboration Application.

4.1.1 Responsibilities

- Monitoring the current session of the user.
- Querying any user updates in the database.
- Displays all the courses the student is enrolled in.
- Recording when the user has gone idle.
- Uploads and removes documents their own documents.
- Can create CourseItems for a specific course.
- Can join or create courses.

4.2 Course

A Course class is able to aggregate all of the information and student created objects that are associated with a particular course. Created objects include all of the CourseItem objects associated with the

Course and a single Chat object to be used as the general chat for the Course.

4.2.1 Responsibilities

- Presenting to the user all CourseItem objects that have been added to the course from all Students
- Presenting the Chat object for the course to allow communication between the Student and all other Students with the chat object
- Querying any updates to the Course objects and updating the CourseItem and Chat objects

4.3 CourseItem

A Course Item class is a reference to an assignment, exam, URL, or other document relevant to the course. The object in code will serve as a container and mainly server metadata information, while the document itself will be stored in the database. The CourseItem is a created object that is created by a user in the Student class and will be shared through a Course's chat object and always accessible on the sidebar. The user who created the CourseItem object will be able to later modify or delete the object.

4.3.1 Responsibilities

- Display to the user the name and description of the courseItem
- Allow the creator of the courseItem to modify or delete the courseItem
- Query any updates to the courseItem and update the courseItem accordingly

- Provide a download option so that a Student can download the contents of a course item

4.4 Chat

A chat is created for each Course and for each CourseItem. Chats are not created by or associated with users in any way, they are only automatically created/exist alongside existing Courses and CourseItems.

4.4.1 Responsibilities

- Tracking and recording to the database new messages in the chat. Messages are written to the database immediately (are not written to a buffer/flushed).
- Notifying the client of changes when the client checks (asks) whether any changes (new messages) have occurred.
- Providing specific messages upon request.
- Caching messages as they are accessed to reduce read operations on the database.
- Displaying active and offline users of chat channel

5 Class Interfaces

5.1 Class Student

An instance of Student represents a user who can create CourseItems and Courses. A student is also able to upload Documents attached to a CourseItem.

5.1.1 Public Constructor Student

Student(String name, String id, Time lastLogin)
Creates a student object containing the name and Case ID.

5.1.2 Public Method GetStudentID

String GetStudentID()
Returns the Case ID of the user.

5.1.3 Public Method GetDurationOfSession

Time GetDurationOfSession()
Returns the length of the current login session of the user.

5.1.4 Public Method IsSessionExpired

Boolean IsSessionExpired()
Returns whether the the current login session is expired.

5.1.5 Public Method CreateCourseItem

Boolean CreateCourseItem(dict options, Course course)
Creates an instance of a CourseItem and returns whether it was successfully made.

5.1.6 Public Static Method CreateCourse

Boolean CreateCourse(String courseID, String CourseName)
Creates an instance of a Course and returns whether it was successfully made.

5.1.7 Public Method GetEnrolledCourses

List<Course> GetEnrolledCourses()
Returns the list of courses a student is currently enrolled in on the application.

5.1.8 Public Method JoinCourse

Boolean JoinCourse(Course course)
Returns whether a student successfully joined to a preexisting course.

5.2 Class Course

5.2.1 Public Constructor Course

Course(String courseID)
Creates a Course object from the supplied courseID. Populates private fields with items from the database queried using courseID.

5.2.2 Public Method GetStudents

List<Student> GetStudents()
Queries the students table of the database to see which students have added this course to their list of courses.

5.2.3 Public Method GetCourseItems

List<CourseItem> GetCourseItems()
Queries the database to return a list of all CourseItems associated with the course.

5.2.4 Public Method GetChat

Chat GetChat()
Returns the instance of the Chat class associated with the Course.

5.3 Class CourseItem

An object created by a student that can contain a document relevant to the course. This object is associated with a Course object.

5.3.1 Public Method RemoveCourseItem

Boolean RemoveCourseItem()
Removes the CourseItem from the database, so that it is no longer available to Students.

5.3.2 Public Method ModifyCourseItem

Boolean ModifyCourseItem(Document document)
Adds the document to the already existing CourseItem.

5.4 Class Document

5.4.1 Public Constructor Document

Course(Integer documentID
Creates a Document object associated with the specified document ID. This ID is used to query the database to locate the actual file associated with it.

5.4.2 Public Method Download

Void Download()
Outputs the binary data for the document along with the proper HTTP header (application/octet-stream) to tell the browser to download the file instead of display it.

5.4.3 Public Method GetCourseItemID

String GetCourseItemID()
Returns the unique id associated with the course item.

5.5 Class Chat

Represents the chat for a course. The Chat object will hold all ChatMessage objects for a course.

5.5.1 Public Constructor Chat

Chat(String courseID)
Creates a Chat object from the supplied courseID. Populates private fields with items from the database queried using courseID. Since there is one chat per course it is ok to use the courseID as a lookup for Courses as well as Chats.

5.5.2 Public Method GetChatVersion

Integer GetChatVersion()
Returns what number message the chat has currently advanced to. This is cached whenever possible and queried often by the user to know when to request updates.

5.5.3 Public Method GetChatMessages

List<ChatMessage> GetChatMessages(Integer number = 50)
Returns that last *number* chat messages associated with this chat (default 50).

5.5.4 Public Method SendMessage

Void SendMessage(String content, Student author)
Creates a ChatMessage object with a string content and sends it to the chat.

5.5.5 Public Method SendDocument

Void SendDocument(Document doc, Student author)
Creates a ChatMessage object with a document attached and sends it to the chat.

5.6 Class ChatMessage

Represents a single message sent in a Chat. This message can be text or a document.

5.6.1 Public Constructor ChatMessage

Chat(String courseID)
Creates a ChatMessage object associated with the courseID.

5.6.2 Public Enum MessageType

Indicates the chat message type. Currently can be:

- Text
- Document

- Image

5.6.3 Public Method GetType

MessageType GetType()
Returns the type of the message.

5.6.4 Public Method GetContent

Void GetContent()
Returns the content associated with the ChatMessage.

5.6.5 Private Method GetText

String GetText()
Returns the text associated with this ChatMessage

5.6.6 Private Method GetURL

String GetURL()
Returns the URL associated with this ChatMessage.

5.6.7 Public Method GetStudent

Student GetStudent()
Returns the Student who created the ChatMessage object.

5.6.8 Public Method GetTime

Time GetTime()
Returns the time the ChatMessage object was sent.

5.6.9 Public Method Delete

Void Delete()
Removes the ChatMessage object from the database. This object will no longer be shown in the Chat object class.

5.6.10 Public Method GetActiveUsers

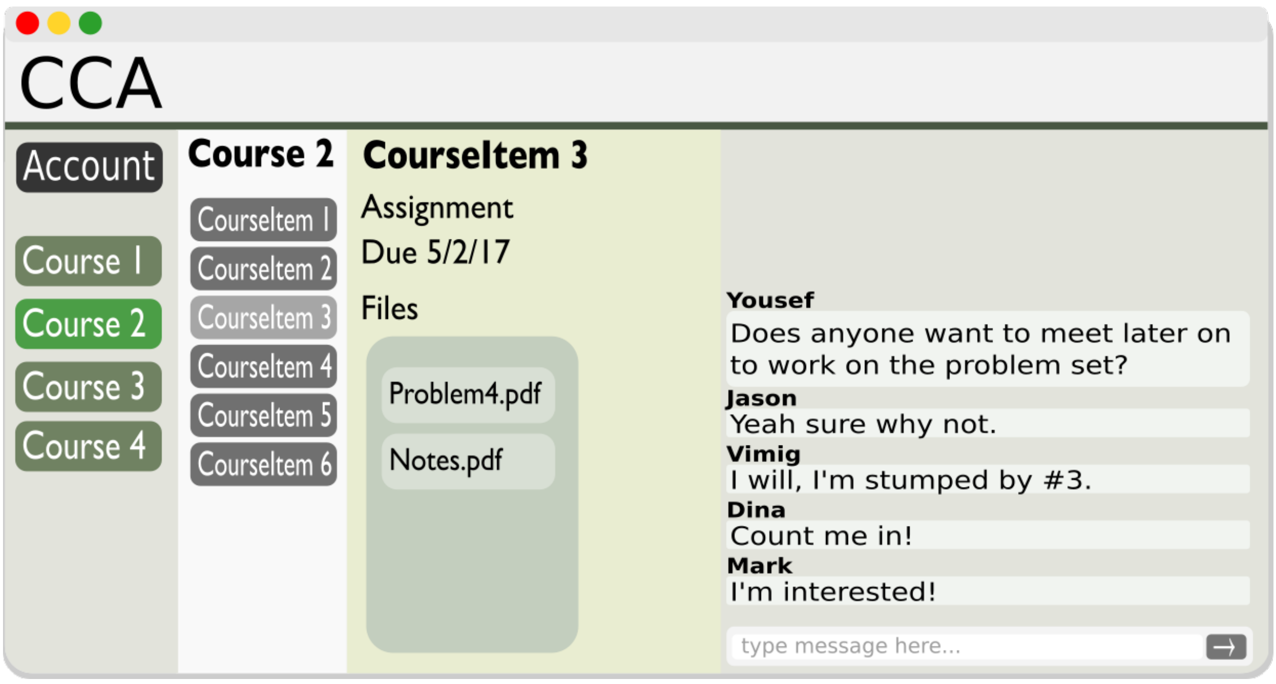
List<Student> GetActiveUsers()
Returns all the active users in the current chat.

5.6.11 Public Method GetAllUsers

List<Student> GetAllUsers()
Returns all users currently subscribed to a chat.

6 Human Interface Design

6.1 Mockup of User Interface



Yousef
Does anyone want to meet later on to work on the problem set?
Jason
Yeah sure why not.
Vimig
I will, I'm stumped by #3.
Dina
Count me in!
Mark
I'm interested!