SOFTWARE REQUIREMENTS SPECIFICATION

Athena : A Course Management System

Version 1.0

Sriyash Poddar (18CS30040)

Software Engineering Laboratory
CS29006

1 Introduction

1.1 Purpose

"Athena", the Greek God of knowledge and wisdom is an online service for the students at any university. A student using this can not only browse details of the courses but also get suggestions regarding the courses he/she can take.The user can check for slot clash between chosen courses, their prerequisites, while at the same time have discussions about them on an anonymous forum. The main purpose for this is to make decision making easier for them.

1.2 Intended Audience and Reading Suggestions

This SRS is for developers, project managers, users and testers. Further the discussion will provide all the internal, external, functional and also non-functional information about "Athena".

1.3 Need

The students at any university have a huge burden resting on their shoulders, which is to prepare for their careers and an important step in that direction is taking academic courses in the university, based on their interests and aspirations. But hundreds of courses,some which are difficult, dependent on others, or even similar put the student into an endless spiral of decision making. So 'Athena' will take care of all of the students needs with respect to managing their courses and making information easily available for them.

"Athena" creates a space for students to know all about the different courses they can take at their university, discuss with other people and for the course instructors to share information about their courses.

The recommender system on the website can suggest courses which can be best taken given the past courses and current interests.

The student can prepare his timetable and check if any courses have similar schedule, or any pre-requisites before his registration.

The website has discussion forums for all the popular courses, where announcements/-media related to the course can be posted anonymously. This will help sharing of syllabus,tips and study material for the course among the peers.

The university can post announcements regarding the course such as any change in schedule, examination announcements or change in the curricula.

1.4 Challenges

The challenges that this system faces is firstly to generate robust and dependable suggestions for the courses. The transfer of information need to be made secure and since, the forums are anonymous, proper modules need to be deployed to ban users which post objectionable content. The media uploaded has to be stored in an efficient way to save server space.The system also has to be made tailor-made for individual universities, suited to their environment.

1.5 Future Work

The recommendation system can be improved by incorporating machine learning paradigms like Collaborative Filtering to generate better suggestions. The search engine for the courses can be expanded to include search on the basis of similar keywords, taking into account semantic analysis. The system can also use Natural Language Processing techniques like Opinion Summarization to generate summaries of reviews of a particular course.

2 Overall Description

2.1 Product Functions

This project will be a Graphical User Interface at the client's end. The user will have to enter no personal data to use the services offered by the website. The user will be able to search for a course on the basis of its code, syllabus, key-features etc. The user will be able to generate suggestions by providing their interests and past courses as keywords. The user can select the courses he has chosen to generate his expected timetable, warnings for course clashes and prerequisite requirements. The user can browse the different forums and notice boards for different courses in a separate tab and download/post content anonymously.

2.2 Operating Environment

This will be a web based system. It will have a server that will actually perform all the information retrieval and will store the course details. The user end will simply be a graphical interface, which will send the input to the backend, to fetch results. The tech-stack to be used is as follows:

The front:

React.js, a JavaScript framework for developing beautiful web applications,along with HTML5 and CSS3 would be used to develop the UI/UX of the platform.

The back:

Flask, a Python framework to create robust backend for web applications.

2.3 Design and Implementation Constraints

Currently, the website has data related to Indian Institute Institute of Technology only. Later, it needs to have a service to customise it for different universities through the GUI directly.

2.4 Tentative Timeline

Week-1: Lab-7	Design
Week-2: Lab-8	(cont.) Design
Week-3: Lab-9	Database Development
Week-4: Lab-10	API Integration
Week-5: Lab-11	Testing
Week-6: Lab-12	Demonstration
Week-7: Lab-13	(cont.) Demonstration

3 System Features

3.1 Searching a Course

3.1.1 Description

This feature is the most basic use of this website. The users will have to use the search tab and provide input to the relevant fields. The search results will be generated on the basis of the input and will also have hyperlinks to the course web page, instructors web page, wiki etc.

3.1.2 Function Requirements

1. Search tab
2. Search for a course by course code
3. Search for a course by by course name
4. Search for a course by by instructor
5. Search for a course by by keyword
6. Button to run search
7. Searching on the database
8. Returning list of courses
9. Ask the user to filter search results by credits, difficulty and time slot.
10. Option to search for pre-requisites of a course in the search results

3.1.3 Database

The data is stored in tables, and the search is done by filtering the table rows on the basis of filters, given by the user.

3.2 Recommendation System

3.2.1 Description

This feature requires a given graph database to built to generate suggestions. Here, the nodes of the graph will be the courses and the edge weight will be their similarity with respect to different features.The user will give multiple inputs and the suggested courses will have links to search the details about them.

3.2.2 Stimulus/Response Sequences

The output will be graphical, showing the courses and how they are linked to each other. The graph will have a specific design to represent the weight of similarity between them visually.

3.2.3 Functional Requirements

1. Accept courses taken by the user
2. Option to select interested fields
3. Option to select interested topics
4. Accept keywords from the user
5. Searching on the database built as a graph
6. Returning list of courses
7. Link to the course when he hovers over the course node in the result graph

3.2.4 Database

The data is stored in the form of a graph with the nodes representing the courses and the weight of the edge between them represents the similarity. Initially information related to curriculum of each course is extracted. Then, the keywords are extracted from the data, and on the basis of matching keywords, department, and field of any two courses the weight of the edge between them in the graph is calculated.

3.3 Building your timetable

3.3.1 Description

This feature is used to build a sample timetable given the user has an idea about what courses the user would like to take. This would give the user an idea about his prospective timetable, and conduct checks for course clashes and prerequisites.

3.3.2 Stimulus/Response Sequences

The output will be tabular, showing the courses added in their respective slots. The allowed courses will be show in green while courses that are not allowed will be shown in red.

3.3.3 Functional Requirements

1. Add courses to the timetable
2. Remove courses from the timetable
3. Show the list of prerequisites when the user hovers over a course
4. Show a warning and a red cell in the timetable for course clashes
5. Button to download the timetable

3.4 Discussion Forums

3.4.1 Description

Each user will be able to add himself anonymously to the discussion forum of any course and share anything related to the course. The user can also receive information from course notice board in the chat rooms.

3.4.2 Functional Requirements

1. Search for discussion forums and notice boards of courses
2. An event is initiated by clicking on the forum link.The code of conduct is displayed.
3. Discussion forum with a notice board, message bar, and chat history
4. All messages appearing chronologically
5. Type text in a text bar
6. Send/Receive text message
7. Button to upload media in a message
8. Media is downloaded when clicked on in a message
9. Send/Receive media
10. Delete unsent message
11. Option to down vote Message

12. User blocked after message has sufficient down votes
13. Button to open the notice board
14. Option to add notice to notice board
15. Option to down vote a notice on the notice board
16. Notice deleted after getting set number of down votes

4 Other Nonfunctional Requirements

4.0.1 Safety Requirements

There is no risk of any threat that can be issued by the system. Since it is a web based application, there will definitely be chances of internet based threats. These need to be minimized as much as possible. A virus scanner will scan all the media before a user can download it. Also if a person is adding a media into the system, then also the document will be scanned for viruses. There is risk for fake news in the notice boards, so adding a down vote option in the notice board controls this. All unruly behavior is punished by blocking user through their IP addresses.

4.0.2 Security Requirements

All the media stored for a particular user will be encrypted by using the standard encryption algorithm.

5 External Interface Requirements

5.0.1 User Interface Requirements

There will exist a Graphical User Interface.Text boxes and button events will be created where ever possible. All errors to be displayed using dialog boxes. The search results will be shown as a list as well as in the form of graphs(if possible). Tabs to use different features will be created. Forums will be designed similar to a generic social messenger.

5.0.2 Hardware Requirements

The server will have a very large memory (5 TB). It should be connected to the internet via high speed cables. It will have decent processing unit (6 cores at 3 GHz average clock speed) and a good graphical processing unit to enable efficient and fast computation of a few modules. The user side need not be that sophisticated. Any device with an internet connectivity and a web browser will be suitable for using the system

5.0.3 Software Requirements

The software requirements are not much for the user. The user should have an access to an internet browser and sufficient space to download media, if needed.

6 Business Policy

6.0.1 Publicizing the product

The use of this product is absolutely free of cost for the students. The application can be customized for other colleges for a charge borne by the college administration. The number of times a feature can be used will limited for a user. Beyond the limit the user will have to pay an additional fee.

6.0.2 Profit

First of all the cost the users will have to pay for the extra features will be low and will aim to only balance the hardware and software costs beared by this enterprise.Moreover, as already mentioned colleges will be interested to purchase it for the benefit of their students. The data of student, including courses, interests etc. is very valuable. This data can be forwarded to several clients including advertising agencies. This data will be made very cheap for those who will be willing to collaborate with us.

SOFTWARE DESIGN DESCRIPTION

Athena : A Course Management System

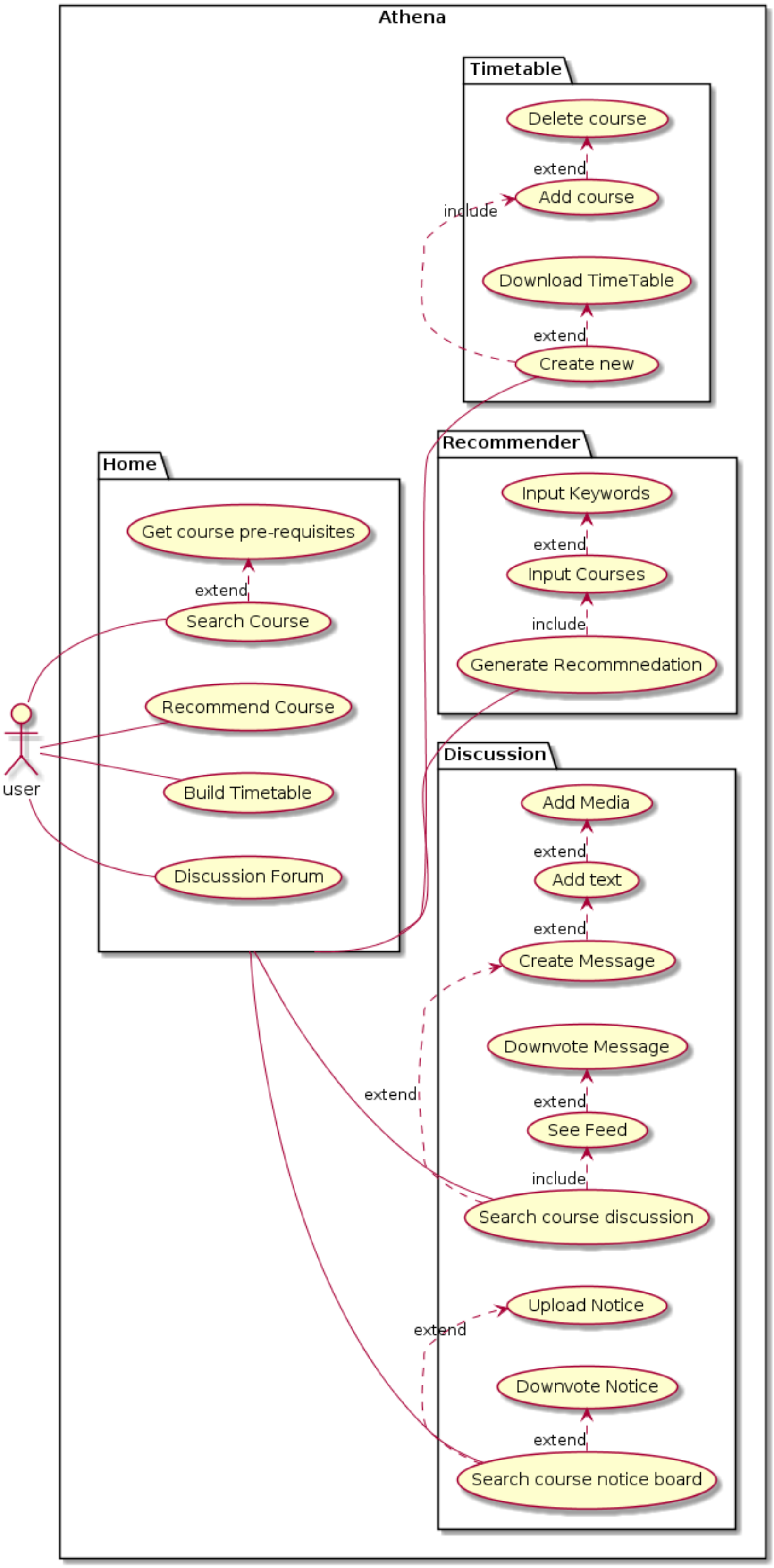
Version 1.0

Sriyash Poddar (18CS30040)

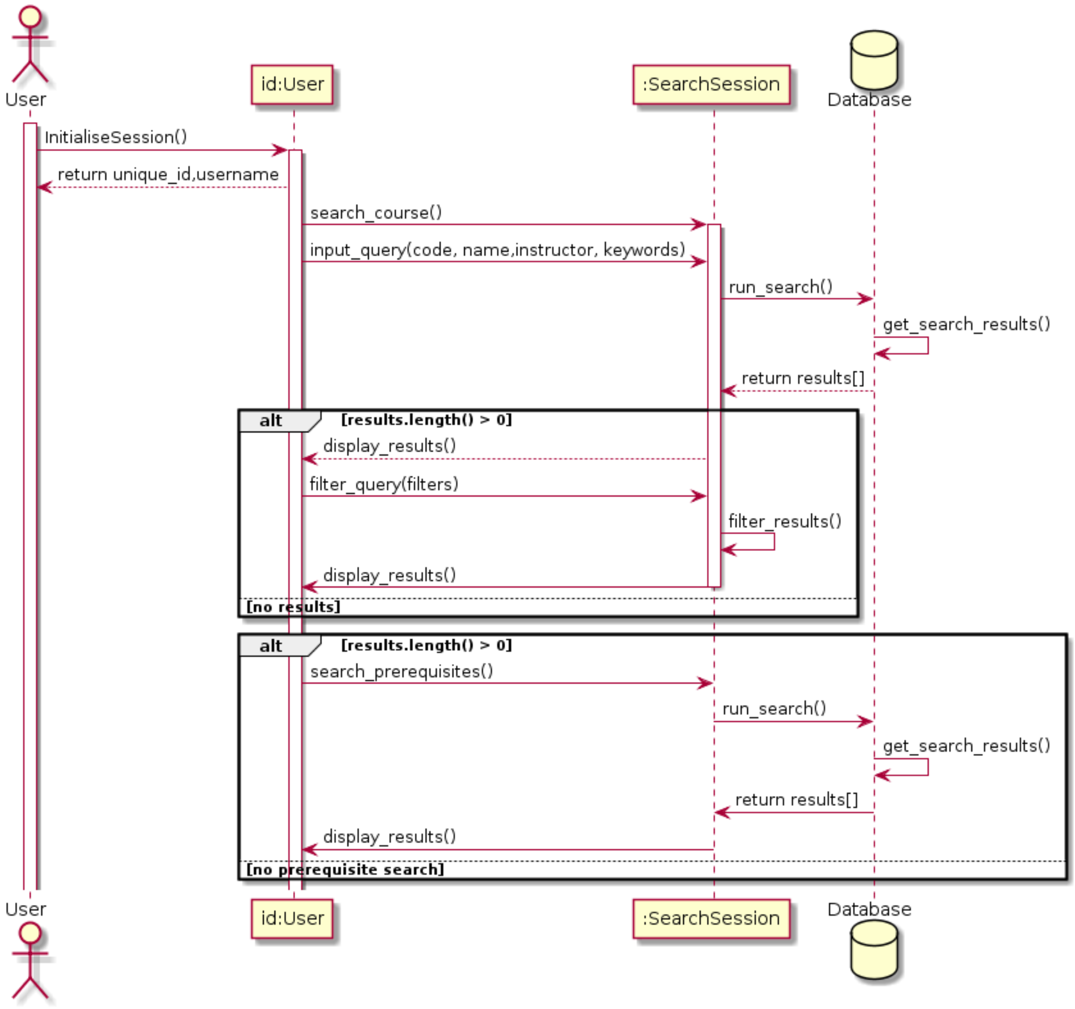
Software Engineering Laboratory
CS29006

Contents

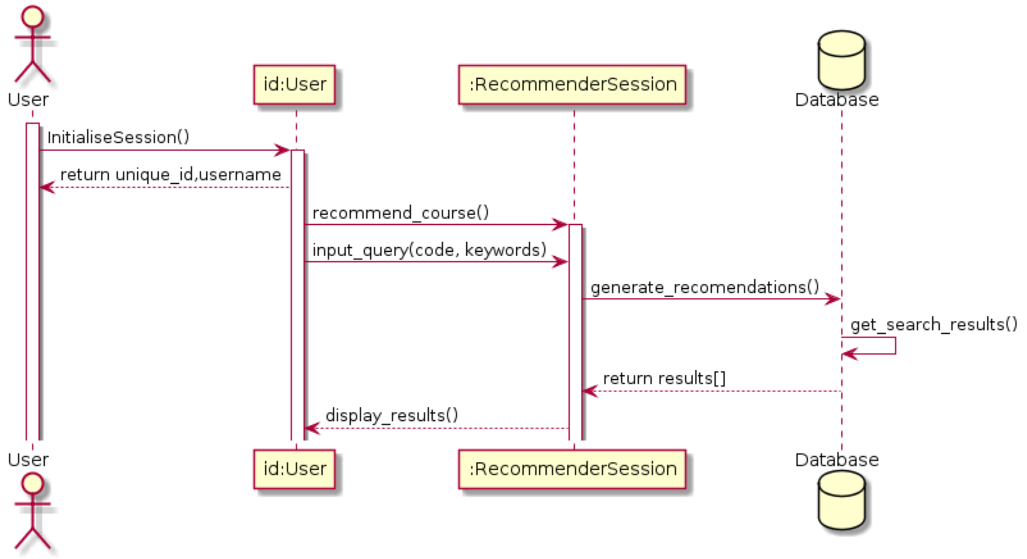
1 Use Case Diagram



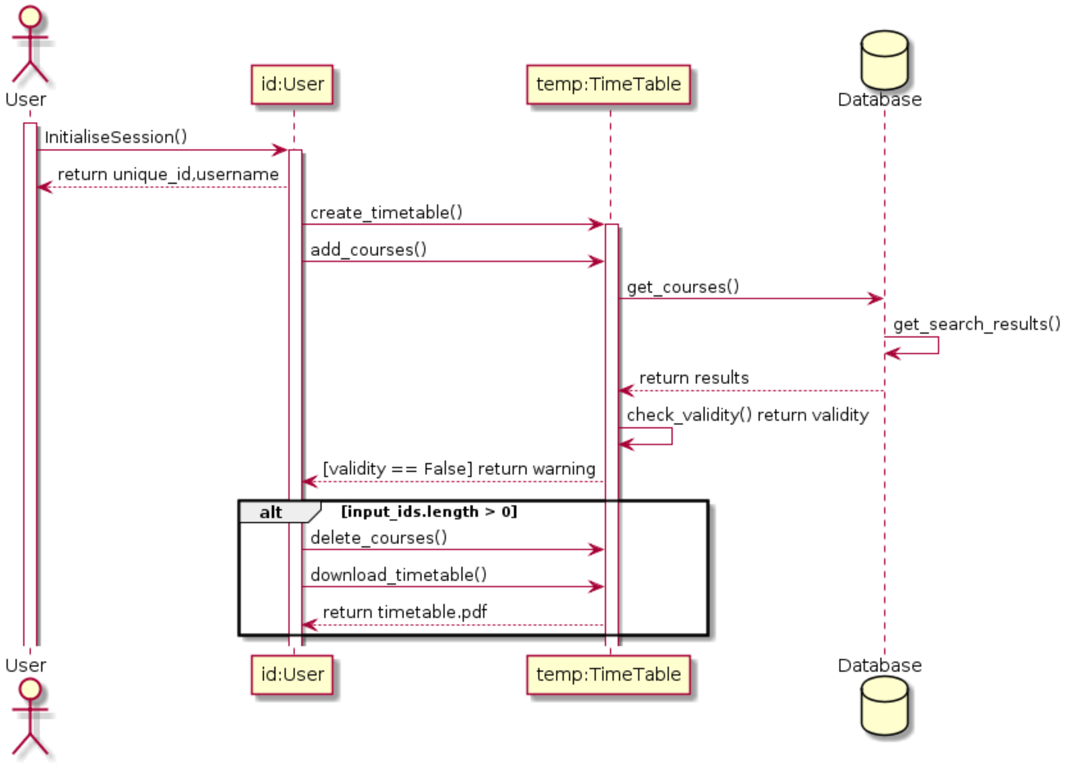
2 Sequence Diagram - Search



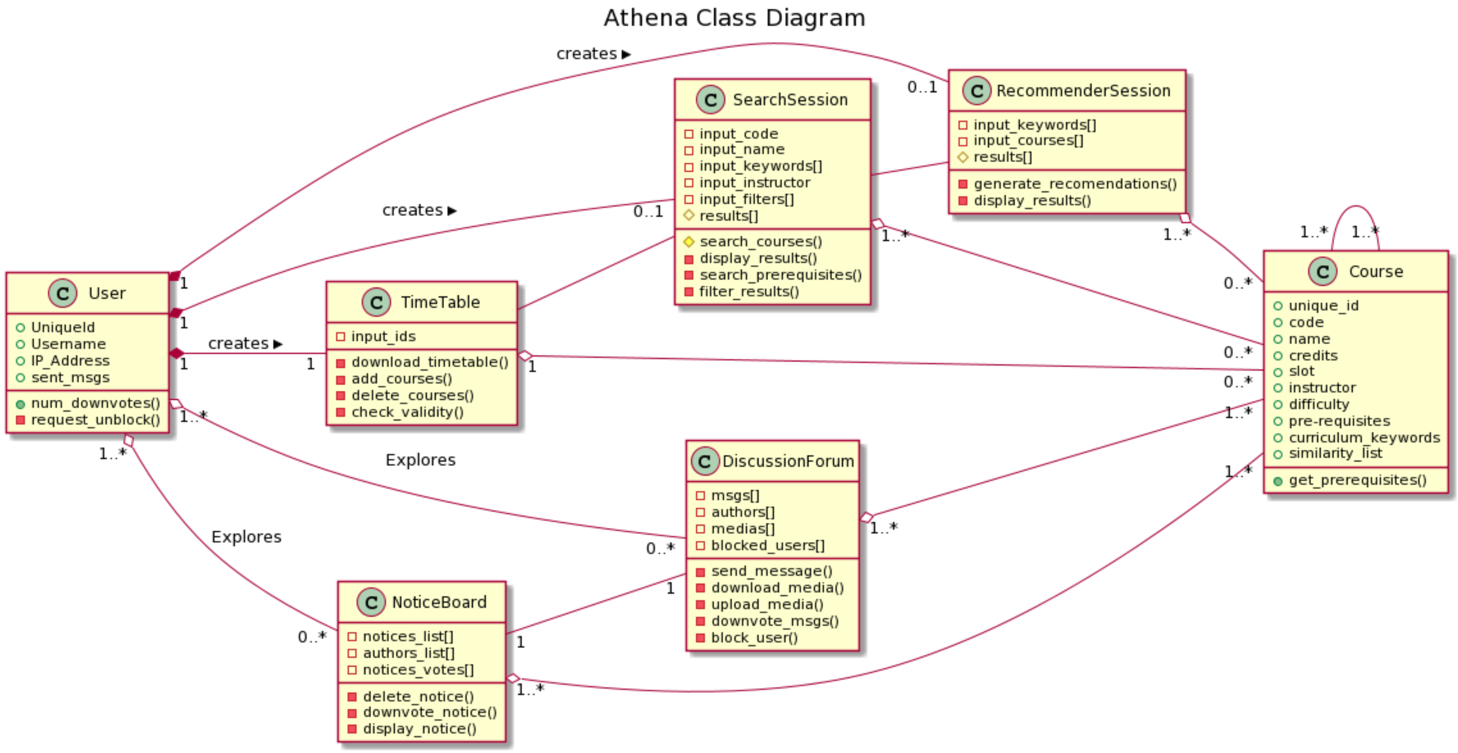
3 Sequence Diagram - Recommend



4 Sequence Diagram - TimeTable



5 Class Diagram


Athena Class Diagram
creates ▸
creates ▸
creates ▸
Explores
Explores
C User
○ UniqueId
○ Username
○ IP_Address
○ sent_msgs
● num_downvotes()
■ request_unblock()
C TimeTable
□ input_ids
■ download_timetable()
■ add_courses()
■ delete_courses()
■ check_validity()
C SearchSession
□ input_code
□ input_name
□ input_keywords[]
□ input_instructor
□ input_filters[]
◇ results[]
◇ search_courses()
■ display_results()
■ search_prerequisites()
■ filter_results()
C RecommenderSession
□ input_keywords[]
□ input_courses[]
◇ results[]
■ generate_recomendations()
■ display_results()
C Course
○ unique_id
○ code
○ name
○ credits
○ slot
○ instructor
○ difficulty
○ pre-requisites
○ curriculum_keywords
○ similarity_list
● get_prerequisites()
C DiscussionForum
□ msgs[]
□ authors[]
□ medias[]
□ blocked_users[]
■ send_message()
■ download_media()
■ upload_media()
■ downvote_msgs()
■ block_user()
C NoticeBoard
□ notices_list[]
□ authors_list[]
□ notices_votes[]
■ delete_notice()
■ downvote_notice()
■ display_notice()
1
1
1
1
0..1
0..1
1..*
1..*
1..*
1..*
0..*
0..*
0..*
1..*
1..*
1
1..*
1..*
0..*
1
1
0..*
1..*
1..*