

Enhancement of Bodhitree through the Development of Additional Features

Avijeet Gaikwad
143050101

Guided by
Prof. Kameswari Chebrolu

Department of Computer Science and Engineering
IIT Bombay

27th June, 2016



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



1 Bodhitree

- Introduction

- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



- Bodhitree is an e-learning platform
 - Design to mimic classroom setting
 - Host multimedia textbooks
- Types of users:
 - Content developer
 - Instructor
 - Student
- Types of courses
 - Bodhi-class
 - Bodhi-book



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Bodhi-class

A flipped classroom model



Source:[1] The University of Texas at Austin: Faculty Innovation Center

Figure : Flipped classroom model



Bodhi-book

A multimedia textbook model

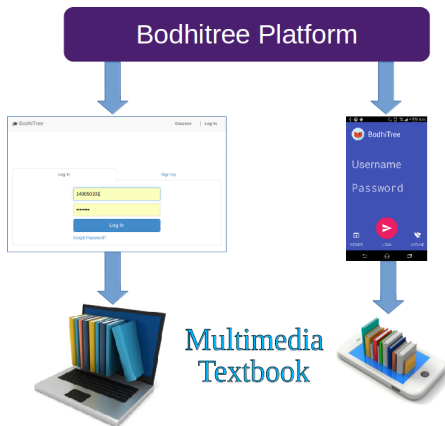


Figure : Multimedia textbook model



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- **Components**
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Overview of the components that constitute Bodhitree

- Courseware
 - Chapters
 - Concepts
 - Progress
- Learning elements
 - Videos
 - In-video quiz
 - Out of video quiz
 - Documents
- Interaction
 - Discussion forums
 - Chat
- Labs
 - Report grading
 - Assignments



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Development details

- Django web framework (Back-end)
 - (M) Designing the model
 - (V) Creating the templates
 - (C) Writing the views and mapping the URLs
- React.js (Front-end)
 - Component based design
 - Virtual DOM



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- **Motivation**
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Objective and Motivation

- Objective is to enhance the Bodhitree platform through the addition of new features
- Need for new features, development ideas:
 - Instructor requirements
 - Student's and Instructor's feedback
 - Brainstorming sessions
- Classification of features pertaining to:
 - Bodhi-class
 - Bodhi-flipped
 - General



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Classification and Overview of the work

- Bodhi-class
 - Marks for offline exams
 - Archiving emails
- Bodhi-book
 - Prerequisites and course graph
 - Access control
- General
 - Facebook style notifications
 - Miscellaneous
 - In-video quizzes ON/OFF
 - Setting importance to threads
 - Email instructors when a new thread is added



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Problem Statement

- Several offline exams are conducted in the Bodhi-class model
- The marks for these offline exams are usually sent by emails, having CSV files attachments
- It is required to have an interface to display these marks to the students on Bodhitree itself



Instructor Specifications

- Upload a CSV containing the students marks
- The file should have the exam details and marks, along with the usernames of the students, in the following format

Students Specifications

- Students must be able to view only their marks that are uploaded for the course



User Interface as seen by the instructor

Upload Marks

Marks File (csv)

Choose File

No file chosen

Upload

Student	Q1 (20)	Midsem (30)	Q2 (20)	Endsem (30)
root	18	25	12	22
avi	12	20	14	24

Figure : Instructor's view of CSV upload and display of students marks



User Interface as seen by the student

Q1 (20)	Midsem (30)	Q2 (20)	Endsem (30)
12	20	14	24

Figure : Student 2's view of his marks

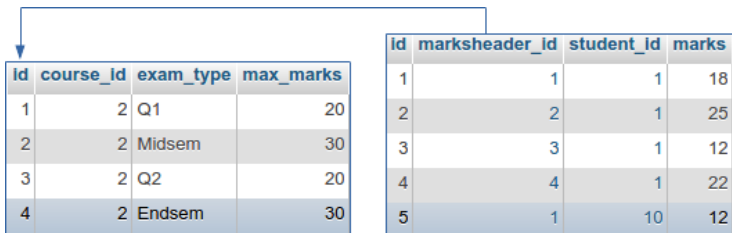


Uploading CSV file

EXAM_TYPE	Q1	Midsem	Q2	Endsem
MAX_MARKS	20	30	20	30
Student 1	18	25	12	22
Student 2	12	20	14	24



Storing marks in the database



Exams details

Marks obtained by students

Figure : Data view of the marks module



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- **Archiving of emails**

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Problem Statement

- Instructors frequently send emails in the Bodhi-class
- These emails are concerned with important announcements and updates in the course
- No easy way for the instructor to view the emails
- Need for archiving and course-wise categorization



Specifications

- All the emails sent by the instructors must be archived
- Instructor should be able to view all the emails sent by him in a particular course
- Student must be able to view all the emails that are received by him
- Sorting and searching features should be available



Archiving the emails

- Function *archive_mail()* called whenever an email is sent
- Foreign keys to the current *user*, *course* and *list of recipients* (many to many relation)
- Postgres automatically adds the current date and time



Displaying the archived emails

- User makes an AJAX request to the server
- Server checks the details of the user in the current session
- Data is fetched in JSON format

```
GET /email_archive/api/getEmails/1/
```

```
HTTP 200 OK
```

```
Allow: GET, HEAD, OPTIONS
```

```
Content-Type: application/json
```

```
Vary: Accept
```

```
{
  "emails": [
    {
      "body": "<p>test</p>",
      "recipients_list": "['avijeet7@gmail.com', 'avijeet@cse.iitb.ac.in']",
      "sender_id": 1,
      "cc_list": "[]",
      "from_email": "instructor_CS224_Computer_Networks@bodhitree.cse.iitb.ac.in",
      "date_sent": "2016-06-14T15:19:06.564Z",
      "reply_to": "avijeetjob@gmail.com",
      "id": 1,
      "course_id": 1,
      "subject": "test"
    }
  ],
  "mode": "S"
}
```



UI for viewing the emails

- Component based design using React.js
- Sorted by most recent first

Sent by: instructor_CS224:_Computer_Networks@bodhitree.cse.iitb.ac.in **Thu Jun 16 2016, 11:40:09 PM**
Subject: [Test Email]Presentations Schedule
Recipients (Click to expand)

avijeet7@gmail.com avijeetjob@gmail.com avijeet@cse.iitb.ac.in

This is a test email.

Sent by: instructor_CS224:_Computer_Networks@bodhitree.cse.iitb.ac.in **Tue Jun 14 2016, 8:49:06 PM**
Subject: test
Recipients (Click to expand)



Search and filter functions

- Searchbox and date filter
- *Instant search*

Search

Start date:

End date:

June 2016

Sun	Mon	Tue	Wed	Thu	Fri	Sat
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2

2016, 11:40:09 PM

Sent by: instructor_CS224:_Computer_Networks@bodhitree.cse.iitb.ac.in

Subject: [Test Email]Presentations Schedule

Recipients (Click to expand)

Sent by: instructor_CS224:_Computer_Networks@bodhitree.cse.iitb.ac.in

Subject: test

Recipients (Click to expand)

2016, 8:49:06 PM



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Problem Statement

- Bodhi-book model lacks significant involvement of instructors
- Dependencies in chapters and concepts
- Proficiency levels required for complete understanding
- Guidelines are necessary for students
- Graphical representation of the course and the guidelines help the students to progress in a better way



Motivation is to ensure that every student has the required background knowledge before attempting to learn a new concept

- Instructor specifies prerequisites for a concept
- Prerequisites viewable on the concept page
- Graph generated using the course data, which serves as a guideline
- Instructor modifies and finalizes the course graph



Storing the prerequisites

- Concepts/Chapters are the prerequisites for other Concepts/Chapters
- Prerequisites are stored as a list of Concept/Chapter id's
- Default values are a blank list `[]`
- In case of a Chapter, a “g” character is appended in front of the id
- Example of prerequisites data for a concept:
`["g1", "5", "6"]`



Design

UI to add prerequisites

- Only the instructor has authority
- Concepts categorized into chapters
- Component based design, immediate request

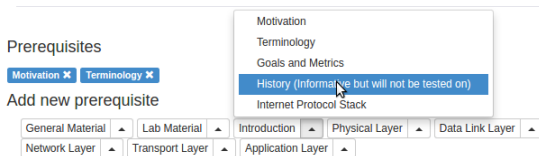


Figure : Adding prerequisite to a concept



Student's view of prerequisites on the concept page

Prerequisites

Motivation Terminology History (Informative but will not be tested on)

- Navigable links



Several graph generation libraries used to generate initial course graphs:

- **Graphviz:** Python library which generates an image of the final graph, non-interactive
- **Arbor.js:** Javascript library which dynamically generates interactive graphs, allows users to interact with the components
- **Vis.js:** Javascript library, interactive graphs, GUI to create/modify existing graphs



Graph generated using Graphviz library

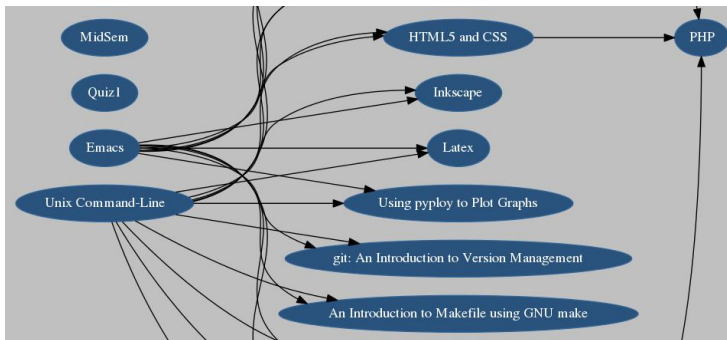


Figure : Part of a complex graph generated using graphviz



Graph generated using Arbor.js

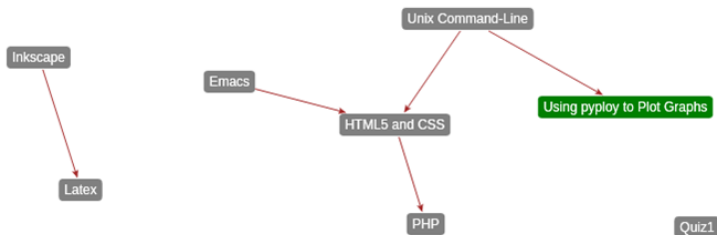
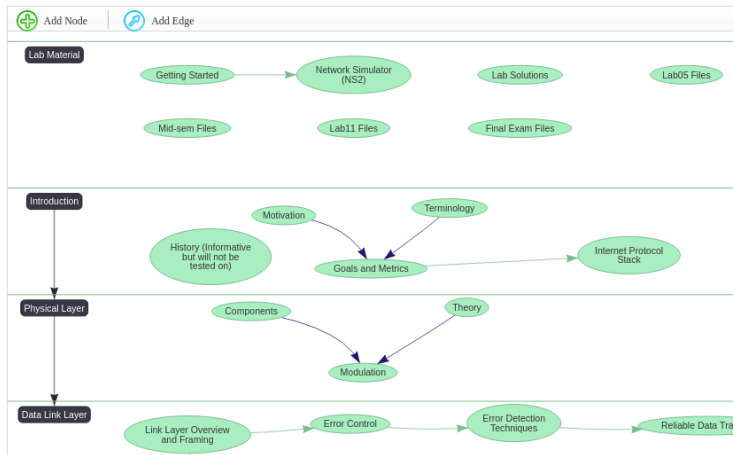


Figure : Part of a graph generated using arbor.js



Graph generated using Vis.js



Design

Creation of graph using vis.js

- Function created to fetch the course data, convert it to JSON and render the graph (html file)
- JSON captured on the client side. Function for creation and organization of nodes and edges written in JavaScript
- Instructor must save the graph before the student can view it
- All the components on the canvas are interactive and modifiable
- Example of Nodes and Edges data:

Nodes	Edges
[{id: 1, label: 'Node 1'}, {id: 2, label: 'Node 2'}, {id: 3, label: 'Node 3'}, {id: 4, label: 'Node 4'}, {id: 5, label: 'Node 5'}]	[{from: 1, to: 3}, {from: 1, to: 2}, {from: 2, to: 4}, {from: 2, to: 5}]



Future work

Ideas to empower course graphs

- Using data from the progress module to display proficiency levels of students
- Generating dynamic, student specific graphs for providing alternative learning paths
- Creating multidimensional graphs, which display in-depth information and provide navigation to courseware elements



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- **Access control**

5 General

- Facebook style notifications
- Miscellaneous

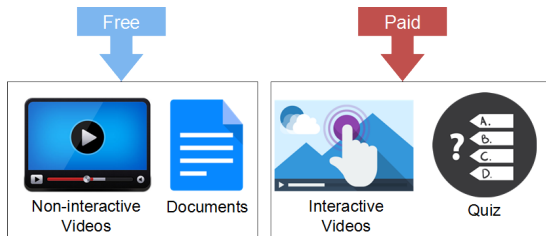
6 Conclusion



Access control

Motivation

Providing differential access to users towards the content



Problem Statement

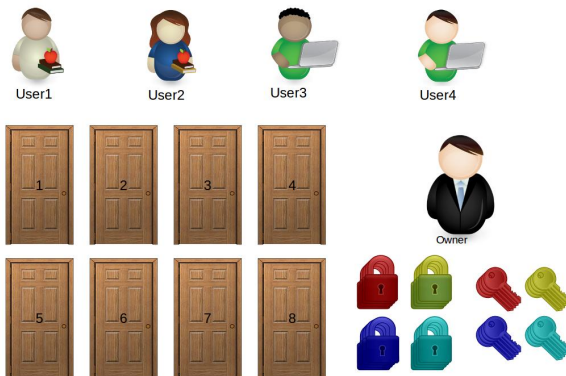
- Giving the instructor authority to limit the access a student has to the content in his course



Specifications

Concept of locks and keys

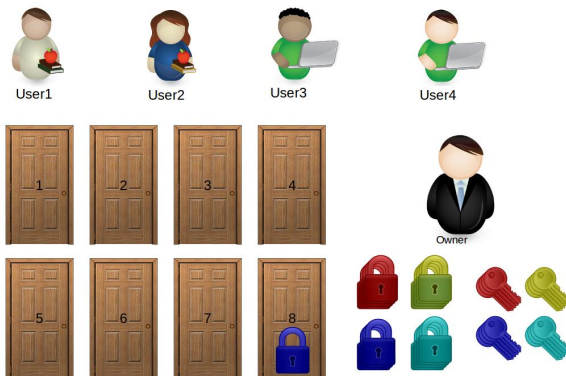
8 doors, 4 users, 1 owner, 4 types of locks and their keys



Specifications

Concept of locks and keys

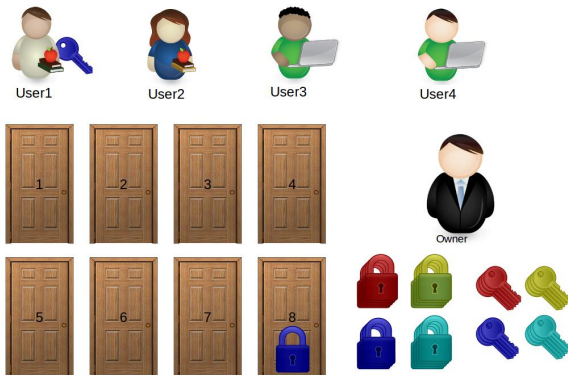
Owner locks door no. 8, thus restricting access to it



Specifications

Concept of locks and keys

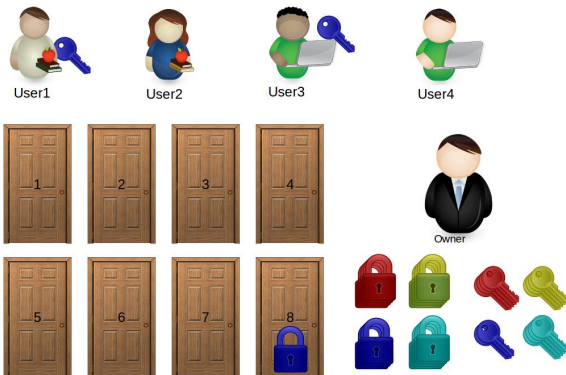
User1 gets access to door 8



Specifications

Concept of locks and keys

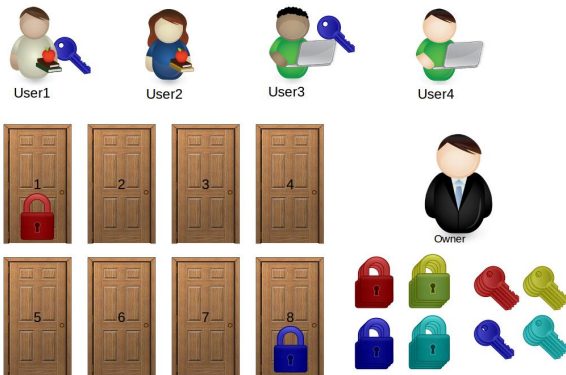
Giving access to other users



Specifications

Concept of locks and keys

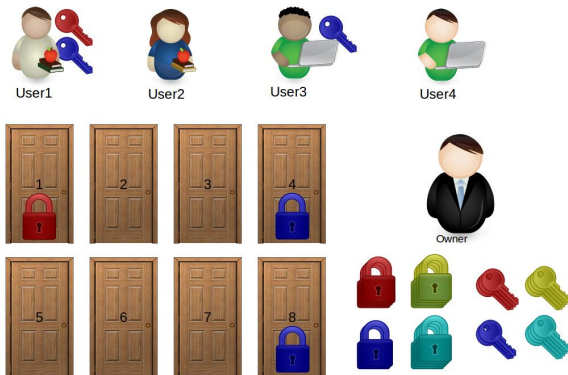
A different type of lock



Specifications

Concept of locks and keys

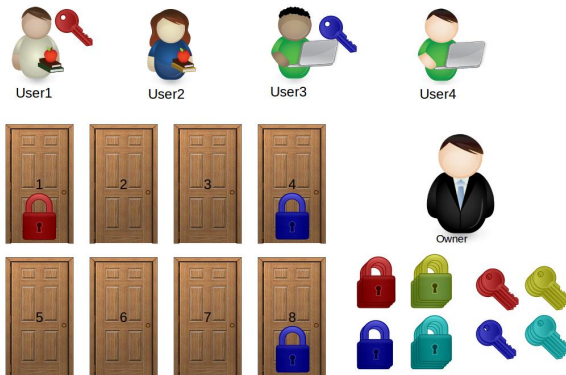
User1 gets access to multiple doors



Specifications

Concept of locks and keys

Restricting access by taking away the key



- Analogy
 - Owner: Instructor
 - Users: Students.
 - Doors/rooms: Content offered on Bodhitree.
 - Locks: Tagging of content by an instructor.
 - Keys: Granting of access to the students for those tags.
- Learning elements that can be tagged:
 - Videos
 - Documents
 - Quizzes
 - Video markers (In-video Quizzes)



- When an element (video, document, quiz or video markers) is created in a concept, the default tag is set as “Free”
- A “Free” tag denotes that all the students who are registered for that course can access that element
- Fields *premium_tag* and *premium_marker* added in the database

avijeet_bt.video_video	avijeet_bt.quiz_quiz	avijeet_bt.document_document
<ul style="list-style-type: none">id : int(11)title : varchar(255)content : longtextupvotes : int(11)downvotes : int(11)video_file : varchar(100)other_file : varchar(100)duration : int(11)premium_tag : varchar(32)premium_marker : varchar(32)	<ul style="list-style-type: none">id : int(11)title : longtextquestion_modules : int(11)questions : int(11)marks : doubleplaylist : longtextis_published : tinyint(1)premium_tag : varchar(32)	<ul style="list-style-type: none">id : int(11)title : varchar(255)uid : varchar(32)is_heading : tinyint(1)is_link : tinyint(1)link : varchar(255)description : longtextplaylist : longtextpremium_tag : varchar(32)



Design

Tagging interface

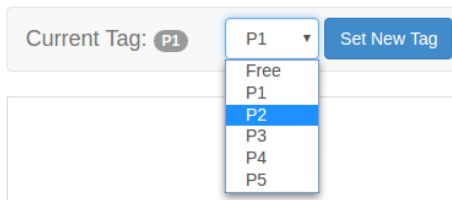
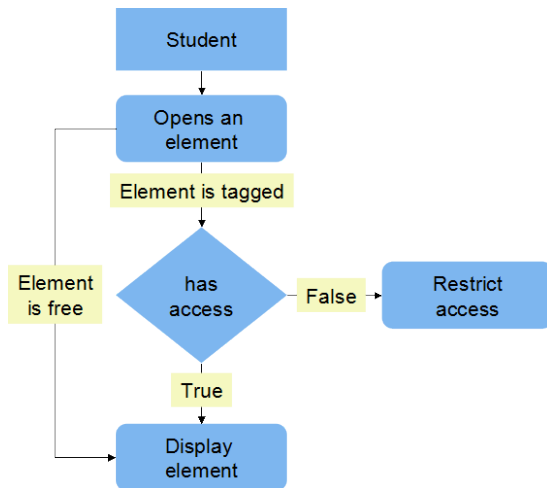


Figure : Tagging interface

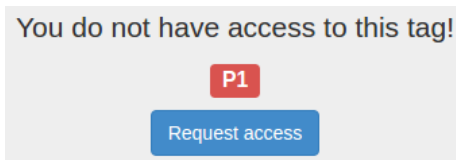
- A badge shows the currently set tag
- A drop down box which lists the 5 fixed tags (P1, P2, P3, P4, P5), and the custom tags
- Setting a new tag changes the *premium_tag* field in the database



Student's access to a learning element



- Server check the user request and sends appropriate data
- Data is not sent to unauthorized users
- The field “*has_access*” is set to *false* in the JSON that is fetched, resulting in the following message



Data sent to client who is not having access to an element

```
"content": {  
  "id": 6,  
  "title": "Components",  
  "content": "The video will cover the details of the components that make up the PHY layer.",  
  "upvotes": 0,  
  "downvotes": 0,  
  "video_file": "",  
  "markers": [],  
  "other_file": "",  
  "duration": 0,  
  "premium_tag": "P1",  
  "premium_marker": "M1"  
},  
"marker_access": false,  
"has_access": false,  
"type": "video",  
"history": {
```



Design

Granting access to users

- Uploading the CSV file:

stud1	P1	P2	P3
stud2	P1		
stud3	P3		

- Checking username against the registered users
- Storing the tags in the database:

id	user_id	course_id	premium_type
10	1	1	P1
5	1	1	P2
8	1	1	P3
4	1	3	P1






Interface to upload the CSV file:

Premium users file in csv format:

No file chosen

List of users who have access to premium features:


Username	Access Types
 root	<input type="button" value="P1"/> <input type="button" value="P2"/> <input type="button" value="P3"/>
 avi	<input type="button" value="P1"/> <input type="button" value="P2"/>
 salman	<input type="button" value="P2"/>






Information displayed after successfully uploading a CSV file

Successfully uploaded the users



Existing premium users:

Username	Access Type
 root	P2

Users added as premium:

Username	Access Type
✓  root	P1 P3
✓  avi	P1 P2
✓  salman	P2

Unknown users:

Username	Access Type
  invalid_user	P1






Removing of tags access from users

- Content developer has the authority to remove the tags access
- Checklist provided listing the users and associated tags
- Corresponding entries removed from the database



Interface to remove tag access

List of users who have access to premium features:

Username	Access Type
<input type="checkbox"/>  root	P2
<input type="checkbox"/>  avi	P1
<input type="checkbox"/>  salman	P2



Creation of custom tags by the instructor

- If the 5 default tags (**P1, P2, P3, P4, P5**) are not enough, the instructor may create new tags of his own
- Once created, custom tags are available in the drop down list of the tagging interface



Interface for adding custom tags

Added tag: Custom2

Add a new custom tag:

Add

List of all the custom tags added:

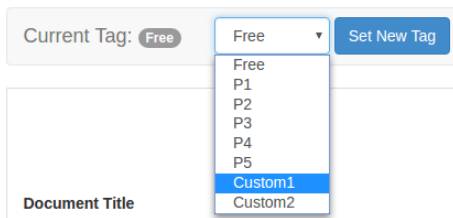
☐ Custom1

☐ Custom2

Delete



Custom tags available in the tagging interface after creation



Design

Group tagging

Tag all the elements (Videos, Documents, etc.) in one click

Group Tagging

Tag all:

Out-Video Quizzes ▼

VideosDocumentsOut-Video QuizzesIn-Video Quizzes and Markers

 as:

Free ▼

Tag all videos

Group Tagging

Tag all:

Out-Video Quizzes ▼

 as:

Free ▼

FreeP1P2P3P4P5Custom1Custom2

Tag all videos



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Problem Statement

- Frequent updates in courseware
- Students may be unaware of the updates
- Need for an interface for quick navigation to the updates



- Notifications must cover the following updates in a course:
 - Publishing a new concept
 - Addition of new videos and documents
 - Publishing new quizzes
 - Uploading marksheet
 - Addition of new threads in discussion forum
- Pending notification must be displayed
- Hyperlinks must be provided for easy navigation



Storing the notifications (PUSH)

- ❶ **Notice** table stores the notification data
 - *course*: Foreign key to the course id
 - *notice_data*: Data displayed in the notification
 - *notice_url*: Hyperlinks of the updates
 - *notice_date*: Date and time
- ❷ **ClearNotice** table stores the notifications clearing time
 - *user*: One to one mapping to the current user
 - *clear_time*: Stores time of clearing the notification

A function **add_new_notice()** has been created which pushes the notifications to the database



Design

Retrieving the notifications

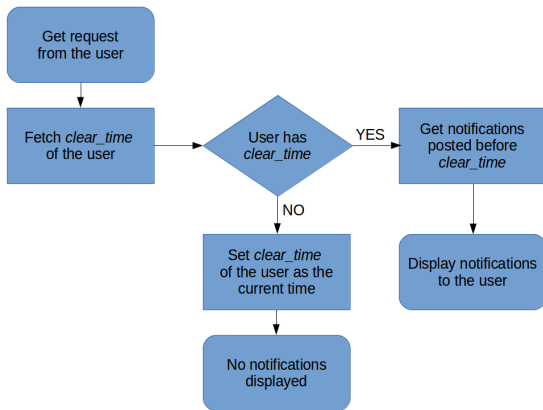


Figure : Flow chart showing the working of notifications retrieval



User Interface

- A badge shows the number of pending notifications
- Drop down list shows the categorized view

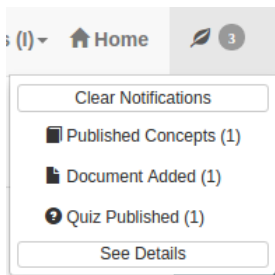



Figure : Notifications indicator along with the drop down list




Detailed notifications view

- Every single notifications is shown in a categorized way
- These notifications can be used for navigation to those updates


[Back](#)[Clear Notifications](#)

 Documents Added

Introduction [Test]Fri Jun 17 2016

 Quizzes Published

Networking Quiz [Test]Fri Jun 17 2016

 Concepts Published

Mid-sem FilesFri Jun 17 2016





1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- **Miscellaneous**

6 Conclusion



- In-Video Quiz ON/OFF
- Setting importance to threads
- Email instructors on addition of a new thread



1 Bodhitree

- Introduction
- Courses: Bodhi-class and Bodhi-book
- Components
- Development details

2 Objective and Overview

- Motivation
- Overview of the work

3 Bodhi-class

- Marks for offline examinations
- Archiving of emails

4 Bodhi-book

- Prerequisites and course graph
- Access control

5 General

- Facebook style notifications
- Miscellaneous

6 Conclusion



Conclusion





The University of Texas at Austin: Faculty Innovation Center

<https://facultyinnovate.utexas.edu/teaching/flipping-a-class>

