Preliminary Project Report

Version 1.0
Prepared by edX development Group
IIT Bombay Summer Intern - 2013
18-06-2013

Revision History

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Reason For Changes</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

# Table of Contents

**Introduction to MOOC (Massive Open Online Course) and edX** .................................................. 3  
  Standard features provided by EdX: .................................................................................. 3  
  EdX User Group and Use: .............................................................................................. 4  
  Structure of EdX: ............................................................................................................. 4  
  Basic Introduction of All Modules: ................................................................................ 5  
  Technologies used: ........................................................................................................ 6  

**Various Modules** ................................................................................................................ 6  
  xqueue .......................................................................................................................... 6  
  Interaction of xqueue with LMS and ORA ..................................................................... 6  
  Ed-Insights .................................................................................................................... 7  

**EdxAnalytics** .................................................................................................................. 10  
  Analytics platform: ....................................................................................................... 10  
  Requirements of Analytics platform for edX system: .................................................... 10  
  EdxAnalytics: ............................................................................................................... 11  

**Ease (Enhanced AI scoring Engine)** .................................................................................. 13  
  Overview ...................................................................................................................... 13  
  Working (Essay Grading) .............................................................................................. 13  
  CREATING MODEL (MODEL CREATION AND TRAINING) ....................................... 13  
  Inputs: .......................................................................................................................... 13  
  Process: ...................................................................................................................... 14  
  Output: ........................................................................................................................ 14  
  Model Creation process in detail: ................................................................................ 15  
  Feature Extraction in detail: ....................................................................................... 16  
  Grading (PREDICTION) ............................................................................................... 16  

**Discern API** ....................................................................................................................... 18  
  Essay Grading Outline: .................................................................................................. 18  
  API Models: ................................................................................................................ 19  

**XBlock** ............................................................................................................................ 20  
  Roles: .......................................................................................................................... 20  
  Layers of Code: ........................................................................................................... 20  
  Architecture: ............................................................................................................... 20  
  State: ............................................................................................................................ 21  

**EdX DATABASE** ............................................................................................................... 21  
  Sqlite: .......................................................................................................................... 21  
  MONGODB: ............................................................................................................... 23  

**Installation Steps** ............................................................................................................. 24  
  Deploying The edX platform ....................................................................................... 25  

**Creating UI on edx-platform** .......................................................................................... 26  

**References** ...................................................................................................................... 28
Introduction to MOOC (Massive Open Online Course) and edX

A **massive open online course (MOOC)** is an online education system providing various courses, aimed at large-scale interactive participation and open access via the web. Apart from traditional course materials such as videos, study materials, and problem sets, MOOCs also aim to provide interactive user forums that help build a community for the students and professors. MOOCs are a recent development in Education system for making distant education possible and effective.

EdX is a non-profit organization formed by collaboration of Harvard and MIT. EdX offers MOOCs and interactive online classes from various domains like (humanities, law etc) from various institutes all over the world.

Development of EdX aims to implement all standard features of EdX MOOC also to enhance it. The final product of this project is expected to be a MOOC based on EdX for IITB and the proposed name for this product is IITBX.

**Standard features provided by EdX:**

- Interactive video lectures with subtitles and indexing on subtitles (Downloadable).
- Study materials like books, notes, cheat sheets etc (Downloadable).
- Online test of different types like video embedded quiz, practice sessions, midterm exam, final exam etc.
- Virtual Laboratory with interactive interface for user to view the expected simulation
- Calendar Based schedule.
- Multi Lingual support.
- Discussion Forums.
- Wiki Edits for implementing collaborative learning.
- Progress reports and other kinds of embedded analytics.
- Different kinds of assessment systems for submitted assignments(subjective). It includes:
  - Peer Grading.
  - Self Grading.
  - Staff Grading.
  - Machine Grading.
- Emails and Notification facilities to registered student.
- Provision of certification.
- Registering and unregistering from a course.
- EdX meetups.
- Contacting authors through mailing.
- Support to a large traffic (Users at particular time).

**EdX User Group and Use:**

This system or platform provides an easy and effective way for students and others to opt for free online courses (fee may be charged in case of certification). The benefits of this is that the traditional learning methods (like classrooms with black boards) are upgraded with online interactive support using computers as a tool to provide all kind of courses in interactive manner.

The main objective is distant learning with effective interactive method and provides self assessments through various kind of test, discussions with the experts and also the facility of online virtual laboratory. Although all the courses provided online provide some common facilities but the grading system and few features may vary according to the institute providing the course. The system also provides the notion of certification (there is possibility that some courses don’t provide it). The goal is not just to provide an efficient platform for online education system but also aims for assessment of this system through analyzing the responses, test and option taken by its user to form a statistic result of the successfulness of the system and upgrading of the same to serve the needs.

**Structure of EdX:**

The EdX is implemented in modular structure and involves the usage of languages: Python, Ruby, Perl, NodeJS and involves database system like MongoDB, Sqlite. It is built on the python framework known as Django- 1.4.5. .

The modular structure of EdX allows the system to be more robust and allows to incorporate system changes easily. The structure of EdX is shown in the diagram above. The various modules aims to implement various features of EdX independently and are integrated together to provide a complete MOOC. In EdX there is 2 subgrouping of the modules here referred as “Main Modules” and “Extra Modules”. The main modules comprises of those modules which are the must requirement of a MOOC and the “Extra Modules” consist of those modules which aims to enhance the functionality and in some cases has the provision to be integrated in future.
Basic Introduction of All Modules:

edx-Platform: This is the main edx platform which consists of LMS and Studio.

Xqueue: defines an interface for the LMS to communicate with external grader services.

cs_comments_service: comments as a service which supports voting and nested comments. And also supports features including instructor endorsement for education-aimed discussion platforms.

CodeJail: This manages execution of untrusted code in secure sandboxes. It is designed primarily for python execution.

edx-insights: Development version of an analytics framework for the edx infrastructure.

edx-Analytics: Analytics platform is for EdX built on ed-insights also it built analytics modules as use cases in the development of ed-insights.

edx-ORA(Open Response Assessor): allows for the assessment of open response problems on the edx platform. It take a submission from an xqueue installation, pass it through machine learning grading, peer grading, and staff grading as appropriate, and return a result to LMS.

Docker: It is a open-source engine which automates the deployment of applications as highly portable, self-sufficient containers.

Discern: This allows anyone to use machine learning based automated textual classification as an API service.

EASE (Enhanced AI Scoring Engine): The EASE repo allows anyone to use machine learning based automated classification. This automated classification can work on both free text (essays, content, etc), and on numeric values.

Configuration: it provides a simple, but flexible, way for anyone to stand up an instance of the edx platform that is fully configured and ready-to-go.

edx-tools: It is a collection of miscellaneous tools for use with the edx platform.

django-wiki: A wiki system with complex functionality for simple integration and a superb interface. Store your knowledge with style: Use django models.
Technologies used

The edX platform is built on a Django-based, Python back end, and relies heavily on Javascript on the client side to minimize load on the servers and network traffic. Assets are generally encoded using XML, but authoring tools shield most instructors from this level of detail.

It uses django because Django follows the Model-View-Controller (MVC) pattern. MVC is way of developing software so that the code for defining and accessing data (the model) is separate from request-routing logic (the controller), which in turn is separate from the user interface (the view). It offers separation of concerns i.e. one can change the view without touching the code of controller or model.

Various Modules

xqueue

XQueue defines an interface for the LMS to communicate with external grader services. For example, when a student submits a problem in the LMS, it gets sent to the XQueue. The XQueue then has the problem graded by an external service like ORA and sends the response back to the LMS.

Interaction of xqueue with LMS and ORA

[Diagram of interaction between LMS, XQueue, and ORA]
Ed-Insights

It is the development version of an analytics framework for the edX infrastructure. It is basically a Python+Mongo+Django framework for creating simple, pluggable analytics based on streaming events. It is a python framework because it involves python language for designing all the modules of the framework. It is a Mongo framework because it involves use of the MongoDB files and data structure for the storing and transportation of the data. It is a Django framework because it is utilizing the various built-in features and functions and libraries of the Django. The basic goal of this framework is to define an architecture for simple, pluggable analytics modules. The basis architecture for the analytic platform can be pictorially represented as:

This architecture denotes a notion of separate analytics server, which is a special kind of server in which all kind of analytic modules resides and it is responsible to provide environment for the processing of the analytics on the raw data. The data to the analytics server is provided basically from the three main source:
- Log files(From LMS server)
- Read Replica on MongoDB
- Read Replica on SQL

Log files are basically a file that list actions that have occurred. The log files are basically transmitted from the LMS server which has all the log entries for the user(student or author) interaction with the system. This log files are transmitted through http or sometimes SNS (Simple Notification service) can also be used for transmission of the time sensitive information to the analytics servers flexibly and reliably.
The other sources i.e the Read Replicas on the various database like MongoDB and sql provides information about the content of the system and the information about the user, course etc.

The Analytic server is provided with its own database to store its information. In actual architecture each analytic platform is associated with its own database. This will be more clear in the architecture of the analytical platform/module which can be pictorially represented as below:

From the architecture we can see that in each analytic modules data is coming from the read replicas and log files. In the analytic modules there are functions to extract the properties of the event, to handle it also to arrange the execution of the process of the log events in either batch, queue basis. It queries the databases for the analytic data, and accordingly generates the appropriate view and discover the interface for the user (like author, student) dashboard according to the authorization of the user, the appropriate view is generated by embedding proper analytics in the dashboard of the user on the basis of the authentication of the user for the analytics. The whole architecture is based on the SOA where SOA stands for Service Oriented Architecture which is basically a style of architecting applications in such a way that they are composed of discrete software agent, that have simply well defined interfaces and are orchestrated though a loose coupling to perform a required function.
It is clear from the architecture that each of the analytics module is having its own databases, here it is 3 (mongoDB, Filesystem abstraction and cache), cache is special in purpose for storing temporally about a log event info which is required for very short duration.

The modules of analytics is not working as the stand alone one in the edX system, it interacts with other modules in the following way:

The interaction between various modules is clear from the diagram. Here loghandlerplus, djeventstream and edx-platform are other components of the edX-platform.
EdxAnalytics:
The Analytic Platform for edX

Analytics platform:
A platform to analyze data based on scientific approach and converting it to meaningful data statistics. Pictorially it can be shown as:

![Diagram of Analytics Platform](image.png)

Requirements of Analytics platform for edX system:
The edX system involves the analytics platform for serving one of the objectives of the formation of edX system and that is the study of the online education system success and response by its user to make it more effective and tailor the system on the basis of its user response. Apart from that the analytics platform also provides the facility for both the students and the instructor to involve the different kinds of analytics embedded in their dashboard according to their rights and authorization. For this there is a concept of secret key for authorizing the analytics. The different kind of analytics is like progress report of the student in duration of the course, the response or progress of the students of particular course meant for the author of the course etc. Hence in this way Analytics platform forms a important component of the edX system. The different modules with specifically deal with the analytics in edX is

- edXAnalytics
- Ed-insights.
**EdxAnalytics:**

This is a component of the edX system, basically a analytic platform which provides the area or platform on which various kinds of analytics can be performed on the feeded log data to produce meaningful statics on this data. It also has its own storage for storing the information being produced by this platform. This platform is built on the Ed-insights platform which in turn is another component of the edX system. It also serves to build the modules which acts like use cases for the ed-insights modules. The edxAnalytics also interacts with the ed-insights for various functions. This can be pictorially shown as:

The basis flow of the edX modules can be represented as follows:
Example of Analytics being calculated by this platform is represented as below:

The various parameters being calculated by the system includes the following:
- Total play count.
- No. of students played
- No. of students replayed.
- No. of student skipped
- Play count per student.
- Watching time per student.

Other types of analytics which this system performs are:
- Analytics on Course: it includes no of students participated, no of student who completed the course etc.
- Student analytics: It include per student analytics like how a student is performing in a particular course or in all course in which the student had registered. It include calculation of parameters like:
  - Grades scored.
  - Problems tried
  - Assignment submitted
  - Modules assessed
  - Materials downloaded etc.
Ease(Enhanced AI scoring Engine)

Overview
The EASE repo allows anyone to use machine learning based automated classification. This automated classification can work on both free text (essays, content, etc), and on numeric values.

Let’s say that you have 10000 user reviews for 15 books (ie “I loved this!”, “I didn’t like it.”, and so on). What you really want to do is use the user reviews to get an aggregate score for each book that indicates how well-received it is. But, in your haste to collect the data, you forgot to get scores from the users. In this case, the text of the user reviews is your predictor, and the score that you want to collect from each user for each book is the target variable.

So, how do you turn the text into numbers? One very straightforward way is to just label each of the reviews by hand on a scale from 0 (the user didn’t like it at all) to 5 (they really loved it). But, somewhere around review 200 you are going to start to get very sick of the whole process. A less labor intensive way is to use automated classification.

If you choose to use automated classification for this task, you will score some reasonable subset of the reviews (if you score more, the classification will be more accurate, but 200 should be fine as a baseline). Once you have your subset, which can also be called a “training” set, you will be able to “train” a machine learning model that learns how to map your scores to the text of the reviews. It will then be able to automatically score the rest of the 9800 reviews. Let’s say you also want to take the user’s activity level into account in order to weight the score. You can add in a numeric predictor in addition to your existing text predictor (the review text itself) in order to predict the target variable (score).

This repo gives you a nice, clean way to do that via convenience functions grade, grade_generic, create, and create_generic.

EASE is only an API, it is used by edX-ORA to carry out machine grading.

Working (Essay Grading)

Essay grading can be done via the “grade” function in grade.py and the “create” function in create.py. Call the create function, and pass in the appropriate data, in order to obtain a created model. That model can then be used in conjunction with the “grade” function to get scores for new text.

The following block diagrams will help in understanding the working and flow of EASE better.

Creating Model (Model Creation and Training)

Inputs:
1. Score list and essay list (training set) + prompt for the essays.
Process:

- chk for whether length of text and score lists are equal or not.
- Decide what algorithm to use (regression or classification)
- this is done by counting the number of unique score points in the score list
- using fn f7 in util_functions.py (regression if # unique scores>5 and classification otherwise)
- Create an essay_set object that encapsulates all the essays and alternate representations (tokens, etc), using model_creator.create_essay_set(text, score, prompt_string)
- Gets features from the essay set and computes error
- using model_creator.extract_features_and_generate_model(A, B)
  - A is essay set created in step 4 and
  - B is the type of algorithm determined in step 2
- returns a result dictionary

Output:

result dictionary {'errors', 'success', 'cv_kappa', 'cv_mean_absolute', 'feature_ext', 'classifier', 'algorithm', 'score', 'text', 'prompt'}
Model Creation process in detail:

Essay and Score lists → Dumps data to a json file → Selects Algorithm → Creates an essay set object and add essays to it

Returns
results = {'errors', 'success', 'cv_kappa', 'cv_mean_absolute_error', 'feature_ext', 'classifier', 'algorithm', 'score', 'text', 'prompt'}

Get cv error → Get classifier → Extracts features

Score Set → Selects Algorithm → Algorithm (Regression or Classification)

Checks if no. of Unique scores > 5

Score and Essay Lists and essay prompt → Creates an essay set object and add essays to it → Essay set object

It also adds additional essays in the set.

self._type = essaytype, self._score = []
self._text = [], self._id = []
self._clean_text = [], self._tokens = []
self._pos = [], self._clean_stem_text = []
self._generated = [], self._prompt = ''''
self._spelling_errors = [], self._markup_text = []

Essay Set And Algorithm → Extracts features → Returns a trained classifier, a trained feature extractor object and cv errors.
**Feature Extraction in detail:**

1.1. Initialize Dictionary from essay set → 1.2. Get mean spelling error and spelling errors per word → 1.3. Get grammar errors (pos tags)

1.4. Extract topical words from essays, (bag_of_words feature) → 1.5. Calculate average index of how "topical" essays are key words/essay length

2.1. Generate bag of words feature → 2.2. Generate length feature (length, word count, etc) → 2.3. Generate prompt feature

2.4. Generate overall feature

3.1. Get GB classifier (scikit-learn) → 3.2. Get cv_error mean and kappa → 3.3. fit features and scores into the classifier

**Grading (PREDICTION)**

![Diagram of Grading Process]

Trained model: Classifier Extractor Prompt Algorithm

Result: {'errors', 'tests', 'score', 'feedback', 'success', 'confidence'}

submission → Grader
Create a new essay set object and add (submission,0) to it

extract features from submission and assign score via the model

Generate Feedback
(Spelling errors, grammar errors, topicality, prompt overlap, )

Return score and feedback
Discern API:
Discern is an API wrapper for a service to grade arbitrary free text response. It allows anyone to use machine-learning based automated textual classification as service.

Its main goal is to provide high performance, scalable solution to help students learn effectively.

Feedback is an important part of this API and for this purpose it is very flexible. Also ease repository is required to use all the functionalities of Discern.

Discern is totally separate from edX-ora. Discern is a standalone API that anyone can use and integrate into their code.

They run on different ports as they are separate things. Its an API wrapper for ease.

ML Grading:
Discern provides ML grading as a part of the API. The ML grading is divided into three main parts:
- **ML Grader**: Calls on the machine learning algorithm to grade a given essay.
- **ML Model Creation**: Provides scripts to generate a machine learning model from given input data.
- **ML Grading Utility Functions**: Provides basic functionalities to check status of models, to get path of model etc.

Essay Grading Outline:

1. Create an organization object & course object.
2. Add essay problems and associate them with a problem.
3. Add essay grade objects that are instructor scored & associate each one with an essay.
4. Model is created which is automatically grade.
**API Models:**

Organization: An organization defines a group of users. Eg:- School, University.

User: User is the basic unit.

Course: Course is a container for problems

Problem: Problems contain meta information for a problem such as prompt, maximum score, text etc.

Essay: Essay is the basic unit of the written work.

Essay Grade: Represents a single grade grading an essay.

Membership: Links between an organization and a user.
XBlock

XBlock is a component architecture by edX.org which is Python language-level Application Programming Interface (API). It generates HTML, CSS, JS that coordinate to form block of content on a page. XBlock’s work together with other blocks to build a complete course and be independent of other blocks, so they can be combined flexibly.

Roles

Block Developer - Author of the XBlock code
Content Author - Author of original content
Course Assembler - Content Editor
Student - User who uses XBlock

Layers of Code

Xblock - Code for the XBlock
Runtime - Code which makes XBlock’s work
Workbench - Specific code we wrote to make everything work (Integration).
Thumbs - Sample XBlock written as a separate installable kit

Architecture

XBlocks do not run by themselves, they run within web applications such as Studio, or LMS, known as runtimes. Each runtime provides services to the XBlock, such as storage, URL mapping, and analytics. The structure of a tree of XBlocks is maintained by the runtime, and is made available to the XBlock through a runtime service. Views are how XBlocks render themselves. Views are invoked by the runtime to produce a rendering of some course content. Their results are aggregated together hierarchically, and so are not expressed as an HTTP response, but as a structured Fragment. A Fragment carries HTML content and resources such as Javascript and CSS needed by the content.
**State**

XBlock state is arbitrary JSON-able data. XBlock state can be scoped on several axes:
- By User: State scoped by user is different for every user in the system.
- By XBlock: State scoped by XBlock can be scoped by various aspects of the XBlock:
  - block usage - the instance of the XBlock in a particular course
  - block definition - the definition of an XBlock created by a content creator (potentially shared across runs of a course)
  - block type - the Python type of the XBlock (shared across all instances of the XBlock in all courses)
- all - all XBlocks share the same data

**EdX DATABASE**

The edX database consists of two database systems – Sqlite (the relational database) and Mongodb (the document based NoSQL database). Both of the database systems are used for different set of works.

**Sqlite:**

This is a relational database system where data is organized into different tables. For local dev environment only Sqlite is used to keep the dependencies simple but for production environments mysql comes into role. Since edx is built on django-platform, thus the sqlite database also inherit the django's legacy database.

The role of Sqlite in edx-platform is to store the user information, user profile, courseware details, course and student modules, enrollment data, sample wiki data, certification, licensing, test center and registration details. These also includes how a courses are organized into groups and also stores the permission roles for groups and special permissions for users etc.

Initially in the django's legacy database there are 9 tables which include user, user_groups, groups, permissions, user and group permissions, django site, session and django content type. And django admin log table is added when you enable the administrator end. So the django's database ER diagram looks like:
And the table details are:
After installation of edx-platform there is one mitx.db file created when the database is synchronized with `django-admin[syncdb]` command. The mitx.db file which is created contains the database tables in sqlite format. There are around 85 tables in total in the edx-platform which also include the 9 tables that are present in django's legacy database. The basic E-R diagram of the edx-platform:

![ER Diagram](image)

So this is the user part how the information is stored for users and course ware data in the back end the next is about how NoSQL database mongodb comes into play.

**MONGODB:**

Mongodb being a NoSql document based database is easy to organize is thus used for storing the course content. It stores structured data as JSON-like documents with dynamic schema (MongoDB calls the format BSON). It has collections which are organised into various documents. Collection differ form table as in relational databases as they don't have fixed set of attributes. Their structure need not be same. The Documents differ from rows as in relational database as they don't give information for all the attributes.

Therefore the course content can easily be organized and put on in bson files. Since each document contains _id information which uniquely identifies a document serves as a primary key.

And for all the data stored in the Mongodb a key-value pairs are created so that optimization can be done i.e. these key-value pair serve as a kind of hashing which
optimizes the retrieval and appending of data in the .bson files. So Mongodb is very much scalable and is used for faster data access.

Using MongoDB as a content store is relatively new (previous architectures used XML-based files from the file system) and it seems like a good match since course ware are well modeled as flexible JSON documents. The use of relational data stores for user-data predated the transition of the content stores and thus was not migrated to a NoSQL platform.

Installation Steps

Firstly we need to download or clone the edx-platform repository from the github.com. The link is shown below:
https://github.com/edx/edx-platform

- Download the source code from github.com
- Go to the edx-platform/scripts/ folder
- Run the create-dev-env.sh in the terminal
- And the installation is done.

Note

1) This script may not run on some machines. The script is mainly for Ubuntu 12.04 LTS version. If you came across any error related to proxy then please ensure that you have mentioned your proxy settings in all the files and also in the git config file.

2) To check whether the script has run completely or not, one should check the edx-all folder in the home directory and it should contain the following folders named data, db, edx-platform and log. If the folders are not found then run the script again.
Deploying The edX platform

Then, every time you’re ready to work on the project, just run

   $ workon edx-platform

To initialize Django

   $ rake django-admin[syncdb]
   $ rake django-admin[migrate]

To update templates

   $ rake cms:update_templates

To start the lms & cms server

   $ rake cms
   $ rake lms[cms.dev]

Or to start Django on a different <port#>

   $ rake django-admin[runserver,lms,dev,<port#>]

For example:

   $ rake django-admin[runserver,lms,dev,<8005>]

If the Django development server starts properly you should see:

   Development server is running at http://127.0.0.1:<port#>/
   Quit the server with CONTROL-C. Connect your browser to http://127.0.0.1:<port#> to view the Django site.
creating UI on edx-platform.

This week I was assigned to create UI of LMS of edx-platform module. This module was a website. So first of all I created a logo and favicon icon for 'iitbx' which was the name given by us to this platform. Logo is shown in 2) and favicon icon which is used in the browser bookmarks is shown in 3)
images 1, 4 & 5 show screenshot of my customized edx-platform. Which shows that I have changed it to iitbx rather than edx.
References:

https://code.edx.org
https://groups.google.com/forum/?fromgroups#!forum/edx-code.

Installation :- http://www.discern.readthedocs.org
Source Code :- http://www.github.com/edx/discern