Software Requirements Specification

for

Course Feedback Portal

Version 1.0 approved

Prepared by Group-8

Indian Institute of Information Technology Chittoor

February 17, 2017

Contents

Revision History

17/02/2017	Prepared Document Draft	V1.0
–	–	–
–	–	–
–	–	–
–	–	–
–	–	–
–	–	–
–	–	–
–	–	–
–	–	–
–	–	–

1 Introduction

1.1 Purpose

The software package is developed from scratch exclusively for the Indian institute of Information Technolgy,Sricity. The course feedback system is a management information system for education establishments to manage student data so that the feedback obtained can be used to assess their quality of instructor.

- It enables the instructors to review how students interpret their teaching methods.
- It also enables the administrators, along with other input, to make summative decisions and make formative recommendations such as identify areas where a faculty member needs to improve.

1.2 Document Conventions

The following conventions have been used in preparing SRS:

- **Admin** refers to the administration of a college and not necessarily the server administration in a website.
- **Bold** text is used to emphasize on important keywords.
- Images are provided along with descriptions either above or below the image.
- The hierarchy of the document contents is as described in the index.

1.3 Intended Audience and Reading Suggestions

This document is created for:

- The Instructors of the course "Information Technology Systems" for their review and monitoring progress of the project.
- Server administrators who would like to understand and extend the project or set it up for their institutions.
- Developers who would like to reuse the source code for their own purposed (The project is released under the GNU GPL)
- The software development team to analyse the requirements.

1.4 Project Scope

The scope of the to-be-developed "Course Feedback System" software package is:

- To provide feedback for the faculty based on which he/she can improve the course;
- To identify problem areas and make the planning of curriculum and study-related development activities more expedient.
- To mould and develop the students and teachers idea of good teaching.
- To obtain information for teaching and personnel work-related decisions.

1.5 References

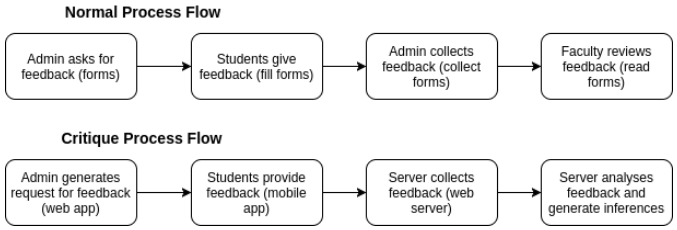
The following references are used in preparing this SRS:

- The SRS template and sample sent by Prof.Nagesh.
- Wikipedia

2 Overall Description

2.1 Product Perspective

Collection of feedback for various courses at the end of a semester is a tedious task for both the students and the administration. Professors also have a hard time going through all this given feedback and trying to figure out where and what went wrong in the course. There is a requirement to make this collection and review process much more seamless and efficient. Our product, Critique, is a web and mobile based application that tries to eliminate the need for manual collection and review of feedback.
Below, we are giving a simple work-flow of how the process of feedback collection happens and how it can be automated using Critique.



2.2 Product Functions

The major functions of the system include:

- Generating feedback request and communicating the same to every student in the system.
- Allowing students to submit forms every feedback request.
- Gathering feedback data from every student in an efficient and automatic manner.
- Analysing the gathered feedback and generating visualisations and inferences for the faculty to easily review the gathered feedback.

2.3 User Classes and Characteristics

The system willl be used by three major user classes. The first class will be the **Admin** class. This includes the college administration whose job is to collect feedback and

submit the gathered feedback to the respective facutlies of respective courses.
The second class of user will be the **Students** class. These users will also be actively involved in the system as they will be providing feedback by filling a feedback form.
The third and final class of users include the **Faculty** class. These users will be passive users as they will not be providing any input into the system. They will only be reviewing the feedback that has been gathered from the students.

2.4 Operating Environment

The operating environment in which the system would operate is divided into two groups. The first environment is the server end, this is a python based server developed in Django. The server would be running on the Ubuntu OS along with the Apache web server for handling requests. The mobile environment would include an Android application that will be used at the students end. Both environments will communicate with each other using the REST API.

2.5 Design and Implementation Constraints

The application will be built for the Android mobile platform. This would limit the users to only those who have an Android based OS running on there phone. The server constraints for the system would be largely hardware dependent. the software used for the server is robust and highly scalable and as such will not impose many constraints on the system. The analysis module is based on the IBM Bluemix server, as such we are only able to utilise a limited amount of these services are free. The Bluemix server will serve a maximum of 1000 requests from a given user for a day.

2.6 User Documentation

Documentation for the various modules would be automatically generated from the source code using Doxygen and given in latex format along with the system.
`http://www.stack.nl/~dimitri/doxygen/`
Other documents for use cases and class diagrams can be found on the github repository.
`https://github.com/chrizandr/ITS_feedback/tree/master/documents`

2.7 Assumptions and Dependencies

An assumption is made that users must know how to operate web applications and mobile applications.
The main dependencies for the mobile application are Android and an active internet connection.

For the server web application, numerous dependencies exists. They are listed as follows:

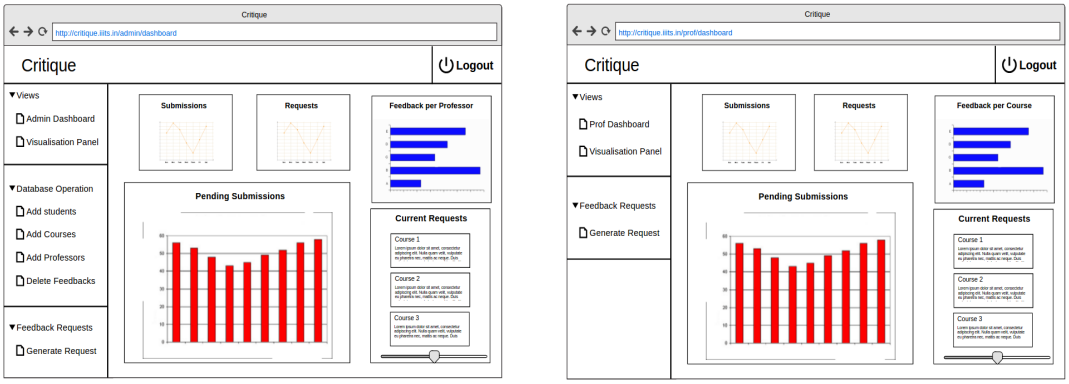
1. Python 3.5+
2. Django 1.10
3. PostgreSQL 9.6+
4. Psycopg2 2.6
5. Virtualenv 15.1

3 External Interface Requirements

3.1 User Interfaces

Basically we have three types of users the professors, admin and the students. The interface for the students is mobile and for the other two types of users it is web interface.

3.1.1 Web-Interface



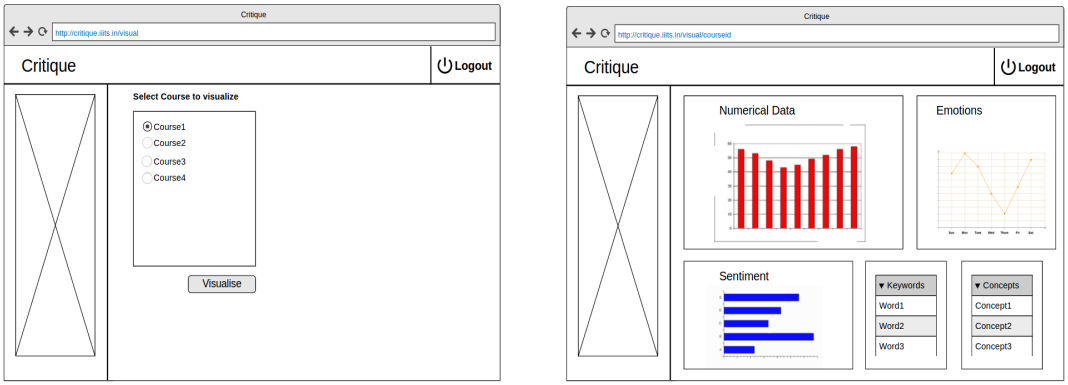
(a) After the admin logs-in

(b) After the professor logs-in

Figure 3.1: User log-in

a)Once the admin logs in, in his dashboard he will be able to see the general data like the number of submissions he has received , the requests that he has made during the past seven days, the positive feedback that each professor got and also the pending requests in the form of graphs. The admin and the professor now have various roles to be performed.
Similarly, the professor also can see the details like the requests he made, the feedback for his courses,pending submissions for his courses etc. The only difference in actions that both these users perform is adding data. Only the admin will be able to add the data into the database.

3.1.2 Visualization

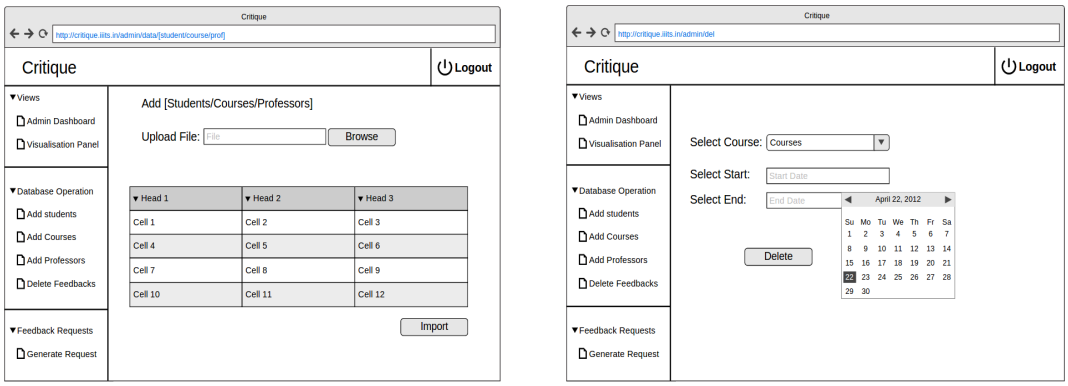


(a) Selecting the course to visualize (b) course visualization

Figure 3.2: Visualization section

b)The first thing he can do is to visualize the data corresponding to a particular course. After he selects the Visualization option he is provided with various courses from which he needs to select the particular course whose data needs to be analysed.Then he will go the visualized data of the course.

3.1.3 Add or Reset data



(a) add data (b) reset data

Figure 3.3: Add or Reset data

c) Here the admin can enter semester-wise details like the courses offered, Student-course pairs, professor-course pairs.If he wants to reset the existing data , he can select a

course and delete the data ranging from start to end date (select the dates from calender option available.)

3.1.4 Request feedback

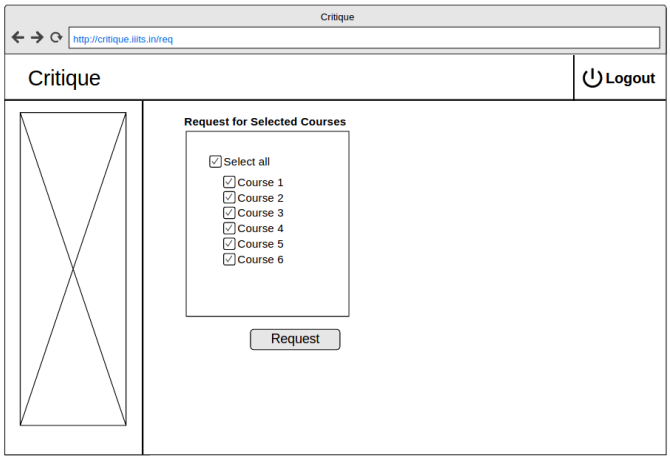


Figure 3.4: Request feedback

d)**Admin**:Admin generates feedback requests for a given course from the students registered for the particular course.He can select as many courses as he wants
professor:The professor can also request feedback for the courses that he offers.

3.1.5 Mobile-interface

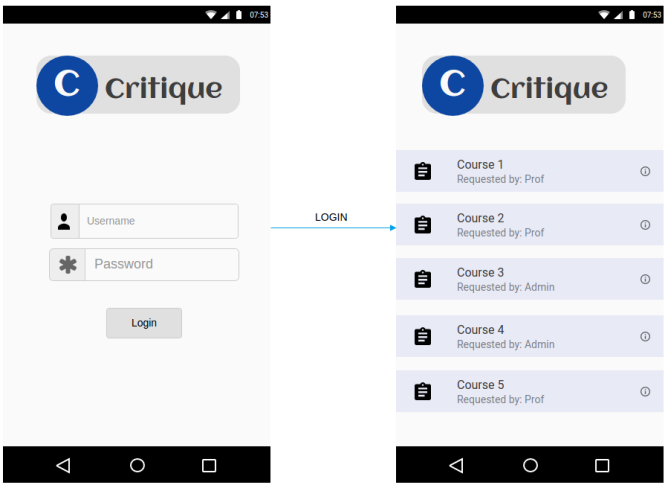


Figure 3.5: Student login

e)The student with mobile app has to log-in with his user-name and password. Then he will be able to see feedback requests for the courses. When he selects a particular notification he will be redirected to a form. Once he uses a notification he will not get another chance to give feedback. By this we ensure that the user can give the feedback only once and only those students get the notification who opted for the course.

3.1.6 Feedback form

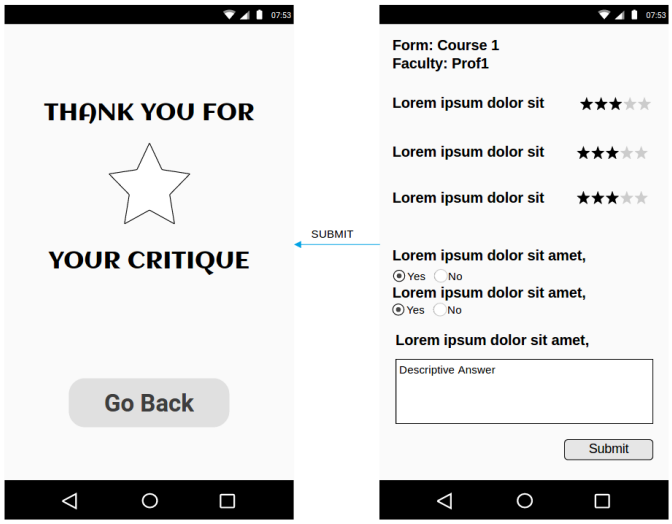


Figure 3.6: Feedback form

f)Here the user fills the user friendly feedback form and submits it. Then he can continue giving feedback for other requested courses.

3.2 Software Interfaces

Dajngo uses an Object Relational Mapping layer to communicate between a PostgreSQL database. Django also uses sockets to serve requests over the HTTP Protocol. Other interfaces include the REST API to communicate between the mobile App and the web server.

3.3 Communications Interfaces

1. **Web browser**: The web browser is the communication interface between the admin/professor and the server. We need active internet connection so that the data analysed appears on our web page. We use the http communication standard.
2. **Android App**: This is the communication interface for the form that needs to be sent to the server (the cloud) for analysis.

4 System Features

The product intends to solve the drudgery of collecting and analyzing feedback. In order to ease the process of collection of feedback we provide following important features arranged according to use-cases :

4.1 Admin and Professor Dashboard

The dashboard provides features for Professor and Admin. The visibility of these features are selective i.e. all features are not available

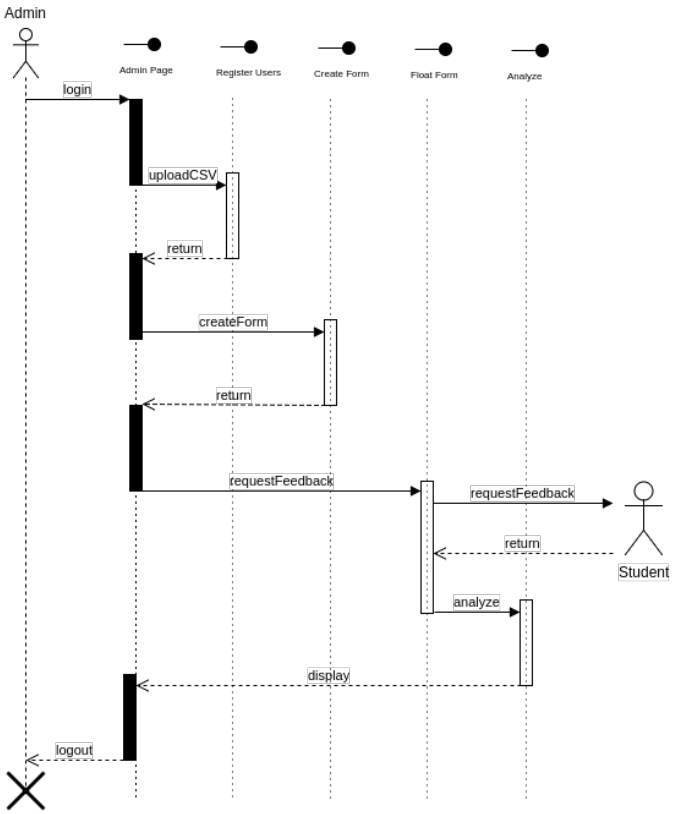


Figure 4.1: Sequence Diagram for Admin User Case

4.1.1 Visualize Data

The product provides the facility for visualizing the aggregate analysis done by the system in intuitive way. For example, the aggregate ratings for each question and the aggregate sentiment for each question needs to be displayed. The visualize feedback is visible to Admin and the Professor.

4.1.2 Request Feedback

For the purpose of requesting feedback, a customizable feedback form is given from which the Professor or Admin can select different question types (ratings or description based questions). Once the feedback is requested the feedback notification is sent to appropriate set of students.

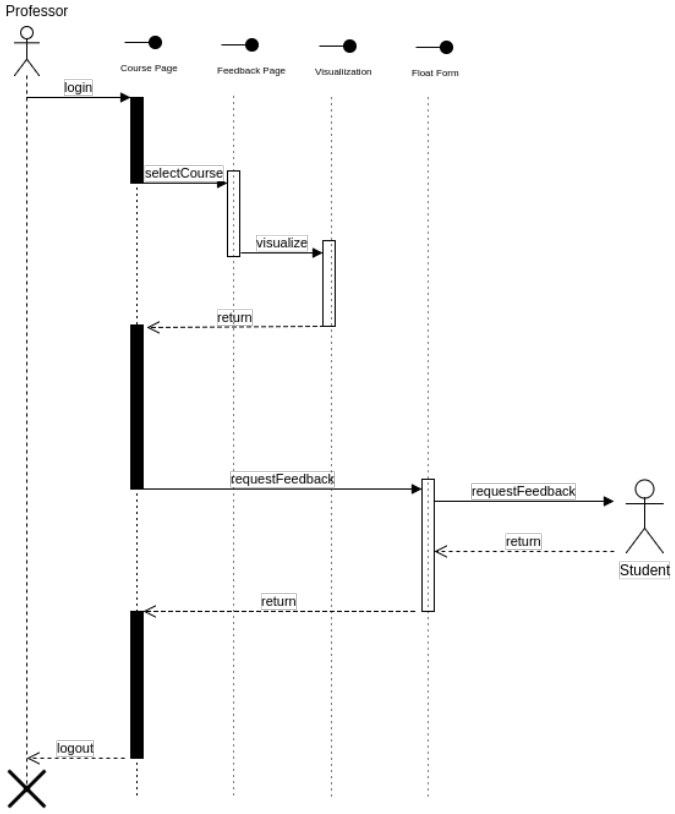


Figure 4.2: Sequence Diagram for Professor User Case

4.1.3 Update Database

The product provides following need-based features where Admin can update the database from a user-friendly UI.

1. Update Students: Upload CSV file for updating models associated with Students.

2. Update Professor: Upload CSV file for updating models associated with Professors.

3. Update Courses: Upload CSV file for updating models associated with Courses.

4.1.4 Reset Database

Reset the database to reset the application state based on need. The expected purpose of this feature is to reset the database at end or start of the semester.

4.2 Student Dashboard

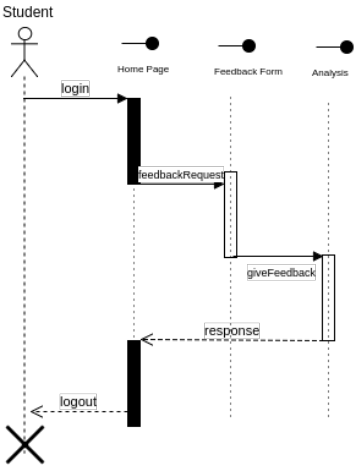


Figure 4.3: Sequence Diagram for Student User Case

The student dashboard is exclusively accessible to student through Android application. The first version of the product is not intended to provide this feature in Web application.

4.2.1 Fill and Edit Feedback Form

For each feedback requested, the student receives a notification to fill th feedback. Student can select the feedback request, edit it and save it for later editing.

4.2.2 Submit Feedback Form

The saved feedback form is submitted given that all the details filled satisfy the feedback form requirements. After successful confirmation, an acknowledgement is sent to student's email as well as Android application.

4.3 Class interactions

The below given diagram shows the interactions between the classes along with data flow:

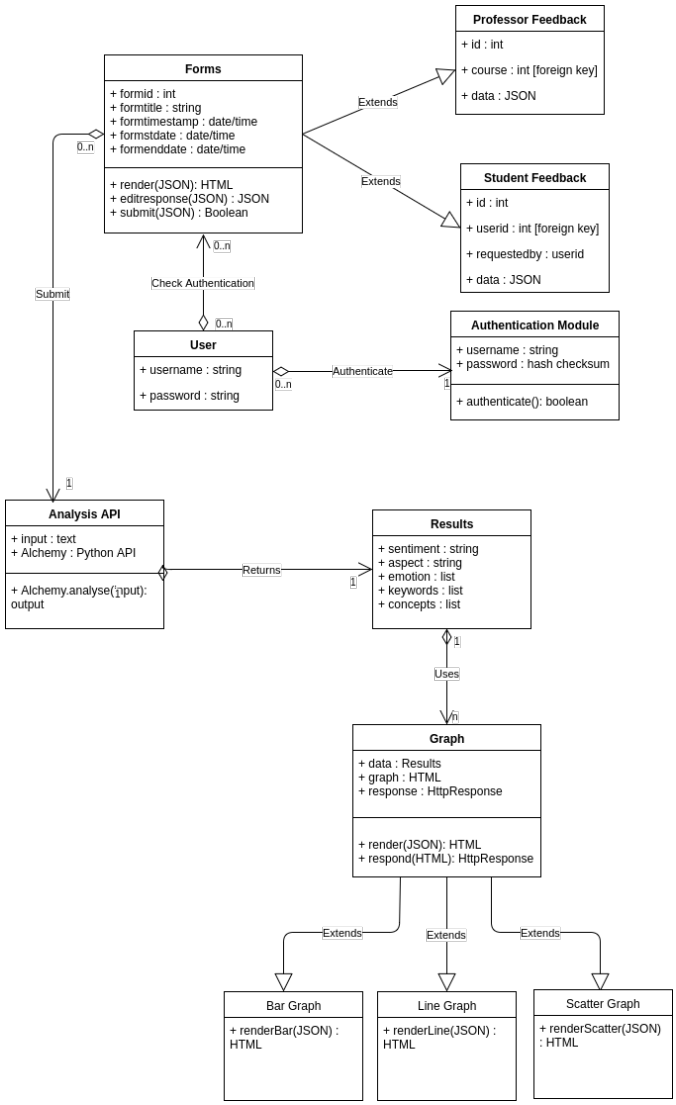


Figure 4.4: Class Interactions

5 Other Nonfunctional Requirements

5.1 Performance Requirements

The software should work for all the courses that the institute can offer. It should be able to handle feedback from around 500-1000 students, where each student will have minimum of 4 courses and maximum of 7 courses. Mathematically, total number of feedback can be around the range of 3500-7000. Even if a course is taken by around 300 students, analysis of the feedback should not take more than 2 -3 minutes.

5.2 Safety Requirements

A student can choose to be anonymous, while giving feedback if he feels that his academics might get affected by his feedback.

5.3 Security Requirements

Only the administrators and instructors should have access to look into the analysed feedback,no student should be able to access it. This software will,
a) Authenticate each user,who logs in.
b) when the user performs any action, authorix=ze him/her to perform the actions allowed for the user and displays error message if found to be unauthorized.

5.4 Software Quality Attributes

5.4.1 Correctness

Tools and the mathematical models utilized to do the feedback analysis should be checked for their correctness. Also, it should be made sure that these integrated models do not corrupt the data from the individual models.

5.4.2 Usability

Graphs and visualizations should be easy and understandable, so that the administrators and instructors can understand the analysis, without putting much effort in learning each graph and its attributes' meanings. The student should not get unnecessary notifications from the server.

5.4.3 Operability

In case of errors and inappropriate access, appropriate error messages should be displayed. Messages should explain how to recover from these errors. Undo and cancel options must be provided for most of the actions. Those actions that cannot provide undo/cancel option, should ask for confirmation from the users.

6 Appendix

6.1 Appendix A: Glossary

6.1.1 Abrrevations and Acronyms

- IBM : International Business Machine
- OS : Operating Systems
- GPL : General Public License
- SRS: Software Requirement Specification
- Prof. : Professor
- HTTP: HyperText Transfer Protocol
- API: Application Protocol Interface

6.1.2 Appendix B: Synonyms

- Prof : Professor, Faculty, Instructor
- Admin : Admin class, Administrators

6.1.3 Appendix B: Database

The descriptive schema for the database is given below:

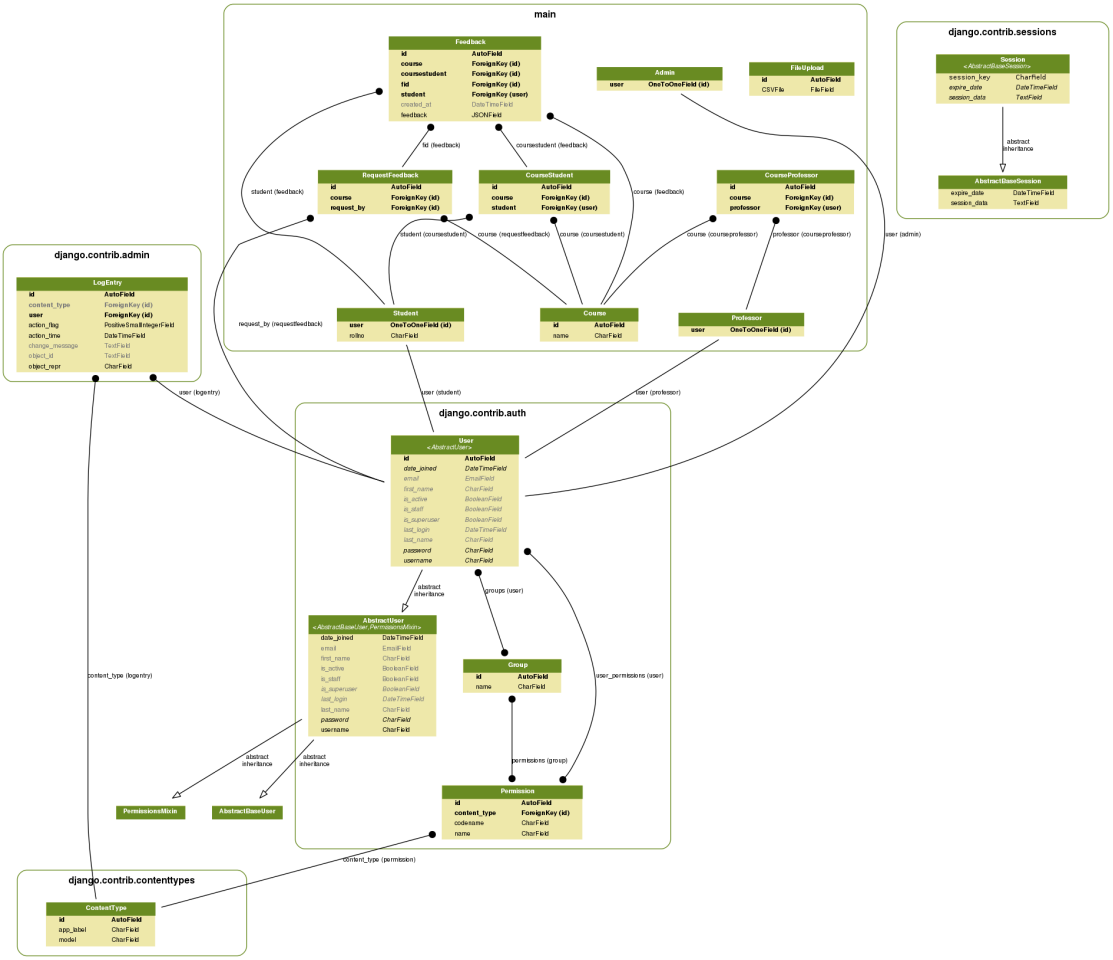


Figure 6.1: Database Schema