Fedena Technical Documentation

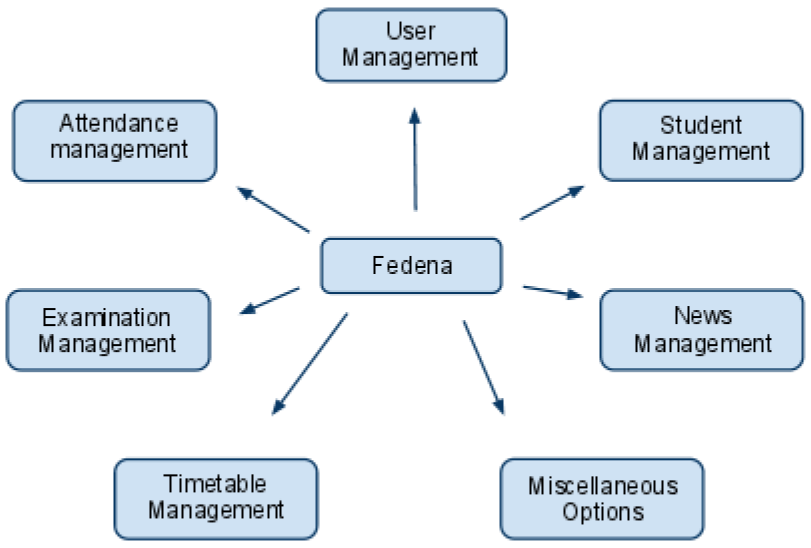
Introduction

This document will explain the logical structure of the Fedena open source school management system.

Basic system modules

The following are the major components of the basic system of Fedena:

1. User management
2. Student management
3. Exam management
4. Attendance management
5. Timetable management
6. News management
7. Miscellaneous settings



User management

The user management module handles the authentication and authorization of users for different pages.

Models and databases

Associated models

The following models are associated with the user management module.

- User

Database structure

Table name: users

- **username** (string) - Username is "admin" for the super-admin that is created automatically, admission no. for students and employee ids for employees.
- **first_name** (string)
- **last_name** (string)
- **email** (string)
- **admin** (boolean) - this is set for admin users (usually school principals, managers, etc)
- **student** (boolean) - this is set for user accounts created for students.
- **employee** (boolean) - this is set for employee user accounts (usually teachers who have less privileges than admins)
- **hashed_password** (string) - SHA1 encrypted password
- **salt** (string) salt string for password.
- **reset_password_code** (string) - used for password resets.
- **reset_password_code_until** (timestamp) - validity of reset_password_code is set when the code is generated and emailed to the user.
- **created_at** (datetime)
- **updated_at** (datetime)

Controllers and views

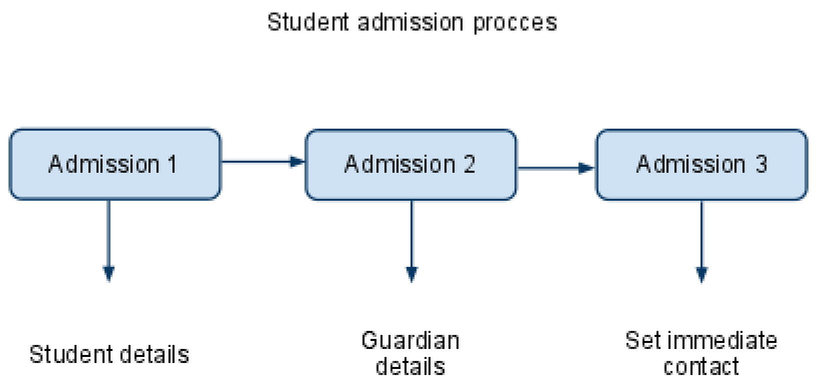
The following controllers are associated with the user management module.

- UserController

Views/methods UserController

- Login (root path) - Login form and link to forgot password.
- Dashboard */user/dashboard* -- Display options acc to user permission levels.
- Manage users */user/index* -- User ajax search.
- View all users */user/all* -- List of all users.
- Create new user */user/create* -- User creation form.
- User profile -- */user/profile/:username*
- Edit user details -- */user/edit/:username*
- Set user privileges -- */user/edit_privileges/:username*
- Reset password page -- */user/reset_password/:reset_password_code* - Verifies if the reset code matches that in the database and allows user to enter new password.
- Delete user account -- */user/delete/:username* - No separate view for this method. Deletes user and redirects to user search page.

Student management



Models and databases

Associated models

- Student
- Guardian

Database structure

Table name : students

- admission_no (string) - A string representing the student's admission no.
- admission_date (date) -The date the student joined the school.
- roll_no (string) - Roll no within the student's class.
- first_name, middle_name, last_name (string) - Student name.
- course_id (integer) - Foreign key reference to the course the student is currently studying.
- date_of_birth (date)
- gender (boolean) - true represents male, false represents female.
- blood_group (string)
- birth_place (string)
- nationality_id (integer) - Foreign key to countries table.
- language (string) - The student's primary language.
- religion (string)
- category_id (integer) - Reference to student category table.
- address (string)
- city (string)
- state (string)
- pin_code (string)
- country_id (integer) - Reference to countries table.
- phone1 (string)
- phone2 (string)
- email (string)
- photo_filename (string) - Filename of uploaded photo.
- photo_content_type (string) - File type of uploaded photo.
- photo_data (binary) - Photo data is stored in database.
- status (string) - Tells us if the student is "Active" or "Former".

- status_description (string) - Description
- immediate_contact_id (integer) - Reference to guardians table.
- created_at (datetime)
- updated_at (datetime)

Table name : guardians

- student_id (integer) - Foreign key to the the student record.
- name (string) - Guardian's name.
- relation (string) - Represents the relation between student and guardian (father, mother etc)
- email (string)
- office_phone1 (string)
- office_phone2 (string)
- mob_phone (string)
- office_address (string)
- city (string)
- state (string)
- country_id (integer) - Foreign key to countries table.
- dob (date) - Date of birth.
- occupation (string) - Guardian's occupation.
- income (string) - Guardian's income range if such details are required for records.
- education (string) - Guardian's educational qualifications level.

Controllers and views

The following controllers are associated with the user management module:

- StudentController

Views/methods associated with StudentController

- Admission page 1 - /student/admission1 - The first step in admission of a new student. All student related data are entered here.
- Admission page 2 - /student/admission2 - The 2nd step in admission of a new student. All guardian related data are entered here.
- Admission page 3 - /student/admission3 - Last step in admission of a new student. The emergency contact is selected from the list of guardians entered earlier.
- Student search - /student/index - Ajax search for students.
- Reports - /student/reports/:id - List of academic and attendance reports for the student.
- Results for exam type - /student/exam_report - Results for the student for all subjects in an exam of a particular type.
- Subject wise report - /student/subject_wise_report - Results in all exams for the student in a particular subject.
- Complete
- Previous years' marks overview - /student/previous_years_marks_overview - Graphical display of student's performance in past years.
- Guardians details - /student/guardians
- Add guardian details - /student/add_guardian

- Edit guardian details - /student/edit_guardian
- Send email - /student/email - Allows email to be sent by admin to student and/or guardians.
- View all students - /student/view_all
- Edit student details - /student/edit
- Remove student profile - /student/remove - Options to remove student from database either as former student or deleting complete records.
- Change student to former - /student/change_to_former
- Delete student records completely - /student/delete - Removes all details of the student.
- graph_for_previous_years_marks_overview - to generate graphs for the view. I doesn't have separate view.
- graph_for_student_annual_overview - to generate graphs for the view. I doesn't have separate view.

Exam management

Models and databases

Associated models

- Examination
- ExaminationResult
- ExaminationType
- Grading

Database structure: <model_name>

Table name : examination_types

- name (string)
- primary (boolean) - Indicates if the result of the exam is the primary result for the academic year.

Table name : examinations

- examination_type_id (integer) - reference to examination type
- subject_id (integer) - reference to subject for which this exam is conducted.
- date (date)
- max_marks (integer) - maximum marks for the exam.
- weightage (integer) - the weightage of the result of this exam to the total score of the academic year.

Table name : examination_results

- marks (integer) - the student's score in the exam.
- examination_id (integer) - the exam for which the marks are stored.
- student_id (integer) - The student who marks are stored.

- grading_id (integer) - The value from the grading table representing the grade received by the student for this exam.

Table name : gradings

- name (string) - Name of the grading level.
- min_score (integer) - The min score for which this grading level is applicable

Controllers and views

The following controllers are associated with the user management module:

- ExamController
- ExaminationResultsController

Views/methods associated with ExamController

- create - Setting up a new exam.
- create_examtype - Setting up a new examtype.
- create_grading - Handles XHR for creating new grading level.
- delete - Delete an exam.
- delete_examtype - Deleting an examtype.
- delete_grading - Deleting a grading level.
- edit - editing an exam's details.
- edit_examtype - Edit exam type details.
- grading_form_edit - Handles XHR for editing grading levels.
- index - Shows links to different options.
- rename_grading - Handles XHR for renaming a grading level.
- update_subjects_dropdown - Handles XHR to return subjects belonging to a particular course.

Views/methods associated with ExaminationResultsController

- save - Save the results of an exam for an entire class.
- update_subjects - Handle XHR to update subjects dropdown for a course.
- update_one_subject - Handle XHR to update subjects for a course.
- update_exams - Handle XHR to populate exams dropdown from subject.
- update_one_sub_exams - Handle XHR to update exams list for a course.
- load_one_sub_result - Handle XHR for displaying the class results for the exam.
- load_all_sub_result - Handle XHR for displaying the class results for the exam.
- update_examtypes - Update exam types dropdown fromsubjects.
- view_all_subs - View class results for all subjects.
- view_one_sub - View class results for one subject.
- one_sub_pdf - PDF reports for one subject.
- all_sub_pdf - PDF reports for all subjects of course.

Views/methods associated with GradingsController

- save - Save the results of an exam for an entire class.
- update_subjects - Handle XHR to update subjects dropdown for a course.

Attendance management

Models and databases

The following models are associated with the Student Attendance Management module.

Associated models

- Attendance

Database structure: Attendance

Table name : attendances

- **attendance_date** (datetime)
- **student_id** (integer)
- **subject_id** (integer)

Controllers and views

The following controllers are associated with the Student Attendance Management module:

- AttendanceController

Views/methods associated with AttendanceController

- index : *attendance/index* :Index page for Student Attendances
- register : attendance/register :Page to select date & class and pull the student list.
- register_attendance : Method which saves the attendance data to database.
- report : attendance/report :Attendance report page to select class and pull student list and show their attendance
- student_report : /attendance/student_report/:student_id :Shows individual student attendance report

Timetable management

Models and databases

Associated models

- PeriodTiming
- TimetableEntry
- TimetableWeekDay

Database structure: <model_name>

Table name : **period_timings**

- **name**(string)
- **break**(boolean)
- **start_time**(date_time)
- **end_time**(date_time)

Table name : **timetable_entries**

- **course_id**(integer)
- **timetable_week_day_id**(integer)
- **period_timing_id**(integer)
- **subject_id**(integer)
- **employee_id**(integer)

Table name : **timetable_week_days**

- **name**(string)

Controllers and views

The following controllers are associated with the TimeTable Module:

- ClassTimingController
- TimetableController

Views/methods associated with ClassTimingController

- index : class_timing/index -> A page for setting up the class timings
- edit   : class_timing/edit/:id -> to edit the class timing
- delete : class_timing/delete/:id -> to delete the class timing

Views/methods associated with TimetableController

- **index :** timetable/index **->** index page for timetable with various options
- **select_class :** timetable/select_class **->** page with option to select class to edit/ create timetable
- **edit :** timetable/edit/:course_id -> Page to create/edit the timetable for a course
- **delete_subject** ->deletes the subject from a timetable cell
- **tt_entry_update** -> Updates a timetable cell on the drag and drop of a subject
- **tt_entry_noupdate ->** Cancels the update of timetable cells
- **update_multiple_timetable_entries ->** When multiple cells are highlighted on click, drag and drop of subject into any one cell, will drop and update the subjects in all selected cells
- **view** : timetable/view -> Option to select a course to view its timetable
- **update_timetable_view** -> On selection of a course, pulls up the timetable for that course

News management

Models and databases

Associated models

- News
- News_comments

Database structure: <model_name>

Table name : **news**

- **title**(string)
- **author_id**(integer)
- **content**(text)
- **created_at**(date_time)
- **updated_at**(date_time)

Table name : **news_comments**

- **content**(text)
- **news**(integer)
- **author**(integer)
- **created_at**(date_time)
- **updated_at**(date_time)

Controllers and views

The following controllers are associated with the News management module:

- NewsController

Views/methods associated with NewsController

- **index:** news/index -> index page for news with various options
- **add :** news/add -> Publish school news
- **add_comment :** news/add_comment -> adds comment to th published news
- **all:** news/all -> shows all published news
- **delete:** news/delete -> delete a news
- **edit:** news/edit -> edits a news
- **search:** news/search_ajax -> search for news
- **view:** new/view

Miscellaneous settings/configurations

Models and databases

Associated models

- Configuration
- Course
- StudentCategory
- Subject
- AcademicYear
- Holiday

Database structure: Course

Table name : Courses

- **grade**(string)
- **section**(string)
- **code**(string)
- **academic_year**(integer)
- **created_at**(date_time)
- **updated_at**(date_time)

Table name : student_categories

- **name**(string)

Table name :subjects

- **name**(string)
- **code**(string)
- **course_id**(integer)
- **no_exams**(boolean)
- **max_hours_week**(integer)

Table name : academic_years

- **start_date**(date)
- **end_date**(date)
- **previous_year**(integer)
- **created_at**(date_time)
- **updated_at**(date_time)

Table name :holidays

- **name**(string)
- **start_date**(date)
- **end_date**(date)
- **created_at**(date_time)
- **updated_at**(date_time)

Controllers and views

The following controllers are associated with the Miscellaneous settings/ configurations module:

- ConfigurationController
- CourseController
- SubjectController
- AcademicYearController
- HolidayController

Views/methods associated with ConfigurationController

- **index:** configuration/index **->** index page for configuration with various options
- **settings:** configuration/settings -> general configurations related to attendance, examinations, etc.

Views/methods associated with CourseController

- **create:** course/create -> create a course for the academic year
- **edit:** course/edit -> edit the course created
- **delete:** course/delete -> delete the course

Views/methods associated with SubjectController

- **create:** subject/create -> create a subject for the course selected
- **edit:** subject/edit
- **delete:** subject/delete

Views/methods associated with AcademicYearController

- **index:** academicyear/index -> creates new academic year

Views/methods associated with HolidayController

- **index:** holiday/index -> creates holidays for the academic year
- **edit:** holiday/edit
- **delete:** holiday/delete

<u>**Gems/Plugins Information**</u>

Gems Used:

- Declarative Authorization : For authorization of various modules and sub modules
- Prawn : For generating PDF reports

Plugins Used:

- Prawnto : Which make use of Prawn Gem to generate PDF's
- Open Flash Chart : For generating Graphical Flash reports