Developer Guide for **Beginners**

[**ERPNext** Tutorials]

SOLUFY SOLUTION SIMPLIFY

ERPNext

www.solufy.in | contact@solufy.in

Contents

1. Environment Setup for ERPNext

(To manually install frappe/erpnext here are the steps)

1. Install Prerequisites:

Python 2.7
high level programming language for general purpose programming. Most server-side commands are executed in Python.

MariaDB 10+
MariaDB is an open source relational database management system (DBMS) that is a compatible drop-in replacement for the widely used MySQL database technology.

Nginx (1.10)
Nginx is a web server and we use it to serve static files and proxy rest of the requests to frappe. You can generate the required configuration for nginx using the command bench setup nginx.

Nodejs (v9)
Node.js is an open source server environment
Node.js is free
Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
Node.js uses JavaScript on the server

Redis server (v=3.0.6)
storage of data structure in memory, used as database, cache and message agent

wkhtmltopdf with patched Qt (for pdf generation)
Create your HTML document that you want to turn into a PDF (or image)

sudo apt-get update && sudo apt-get install mariadb-server nginx redis-server python-dev libmysqlclient-dev

2. Installing the Bench Repo

```
git clone https://github.com/frappe/bench bench-repo
sudo pip install -e bench-repo
```

3. Installing the Frappe Bench

bench init frappe-bench --frappe-branch text (master, development and production)
cd frappe-bench

4. Create a new site

You can then install a new site, by the command bench new-site site1.local.
This will create a new database and site folder and install frappe (which is also an application!) in the new site. The frappe application has two built-in modules Core and Website. The Core module contains the basic models for the application.
Frappe is a batteries included framework and comes with a lot of built-in models.
These models are called DocTypes.

Site Structure:

A new folder called school management will be created in the sites folder. Here is the standard folder structure for a site.

```
.
├── locks
├── private
│   └── backups
├── public
│   └── files
└── site_config.json
```

1. public/files are where user uploaded files are stored.
2. private/backups are where backups are dumped
3. site_config.json is where site level configurations are maintained.

5. Add ERPNext apps using get-app command
bench get-app erpnext https://github.com/frappe/erpnext --branch master

6. Install ERPNext apps
bench --site site1.local install-app erpnext

7. Set developer mode
bench --site site1.local set-config developer_mode 1

To create models, you must set developer_mode as 1 in the site_config.json file located in /sites/library and execute command bench clear-cache or use the user menu in UI and click on "Reload" for the changes to take effect. You should now see the "Developer" app on your desk

```
    {
       "db_name": "bcad64afbf",
      "db_password": "v3qHDeVKvWVi7s97",
       "developer_mode": 1
     }
```

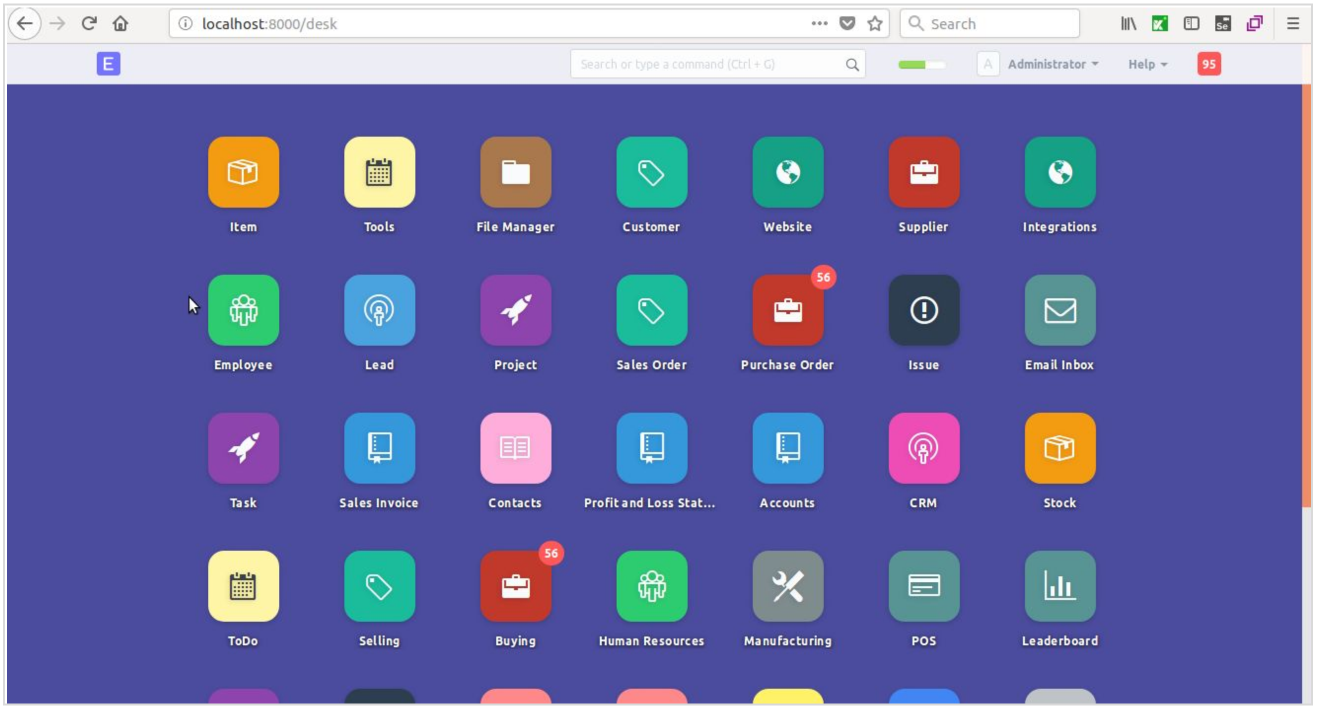
8. Setting Default Site
In case you have multiple sites on you bench use bench use [site_name] to set the default site.
bench use site1.local

9. Bench start

To start using the bench, use the bench start command

Now, execute the bench start command and go to localhost:8000 you can see below screen

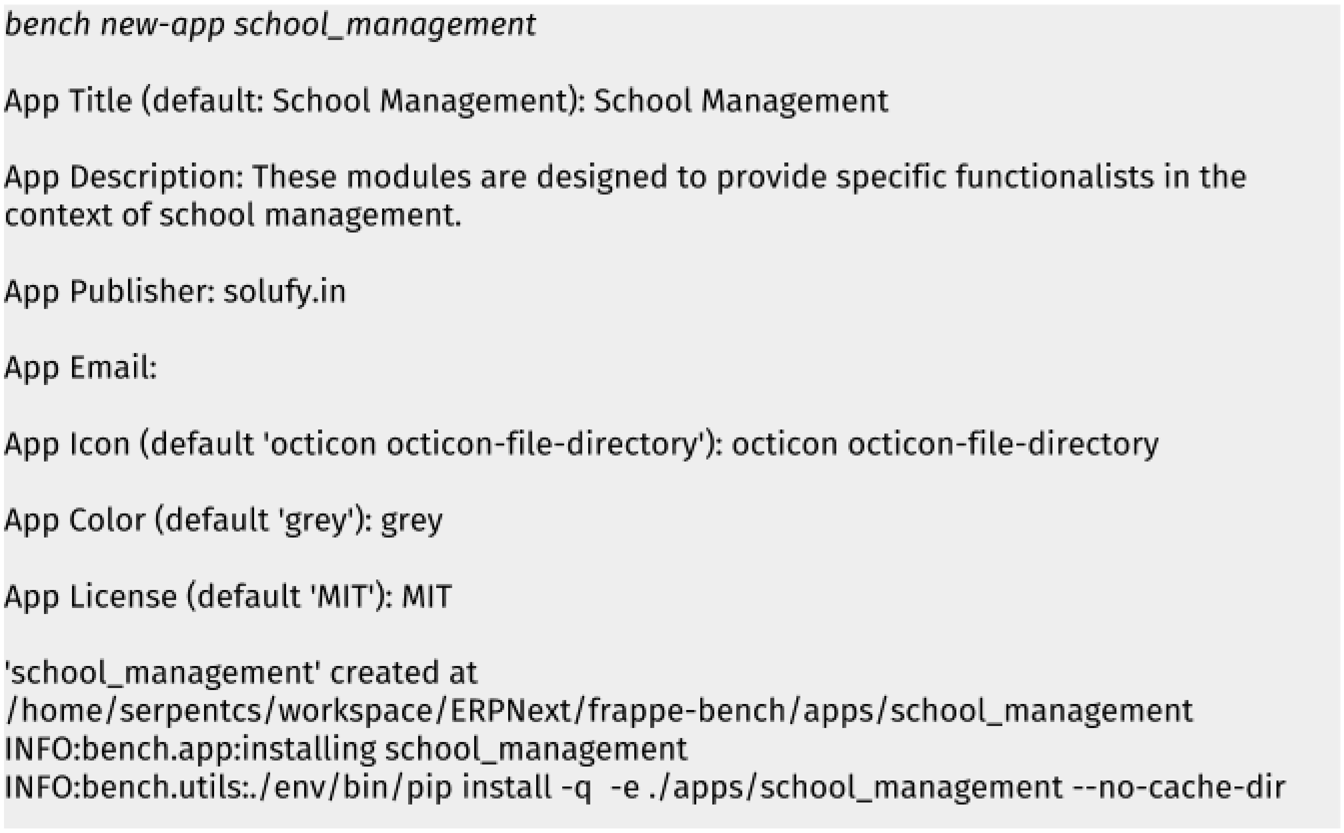
http://localhost:8000



2. Make a school management application

To create our application (frappe-bench folder) execute the command *bench new-app {app_name}* and follow instructions.

The command creates a school management application for you:



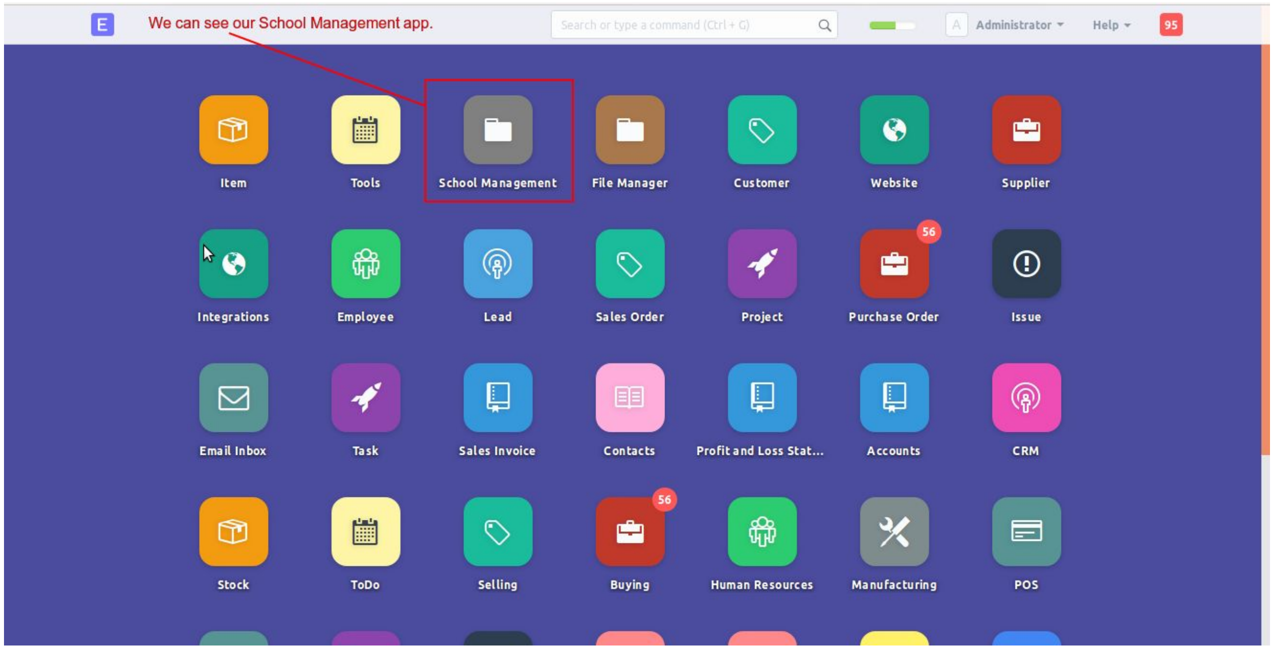
```
bench new-app school_management

App Title (default: School Management): School Management

App Description: These modules are designed to provide specific functionalists in the
context of school management.

App Publisher: solufy.in

App Email:

App Icon (default 'octicon octicon-file-directory'): octicon octicon-file-directory

App Color (default 'grey'): grey

App License (default 'MIT'): MIT

'school_management' created at
/home/serpentcs/workspace/ERPNext/frappe-bench/apps/school_management
INFO:bench.app:installing school_management
INFO:bench.utils:./env/bin/pip install -q  -e ./apps/school_management --no-cache-dir
```

3. Install App

After that install school management app using below command.

```
bench --site site1.local install-app school_management
./env/bin/pip install -q  -e ./apps/school_management

bench migrate
bench clear-cache
bench build
bench start
```

We can see our school management application in below snapshot.

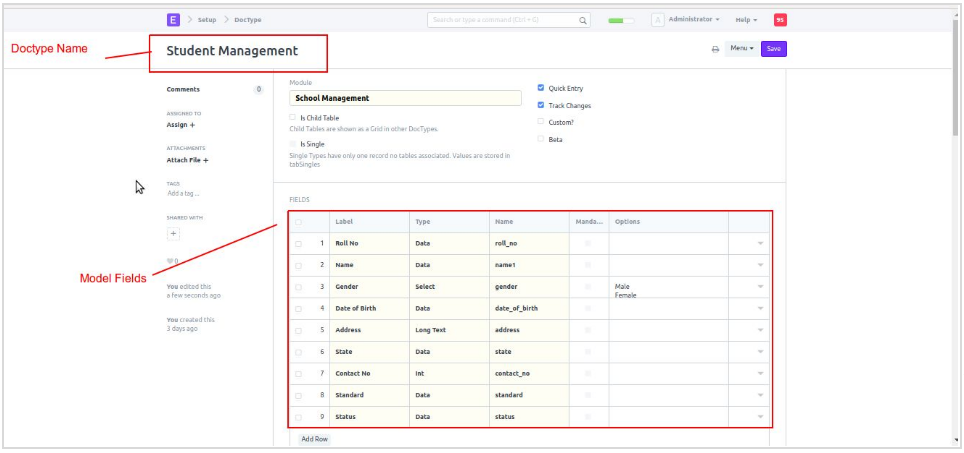


4. Create a docType

We are going to build a simple School Management application. In that application will contents below models:

1. Student Management
2. Fee Management
3. Faculty Management
4. Subject Management

Go to Developer menu > Navigate the documents link > Doctype
Click on “New” button and Create a Student Management doctype



Student Management

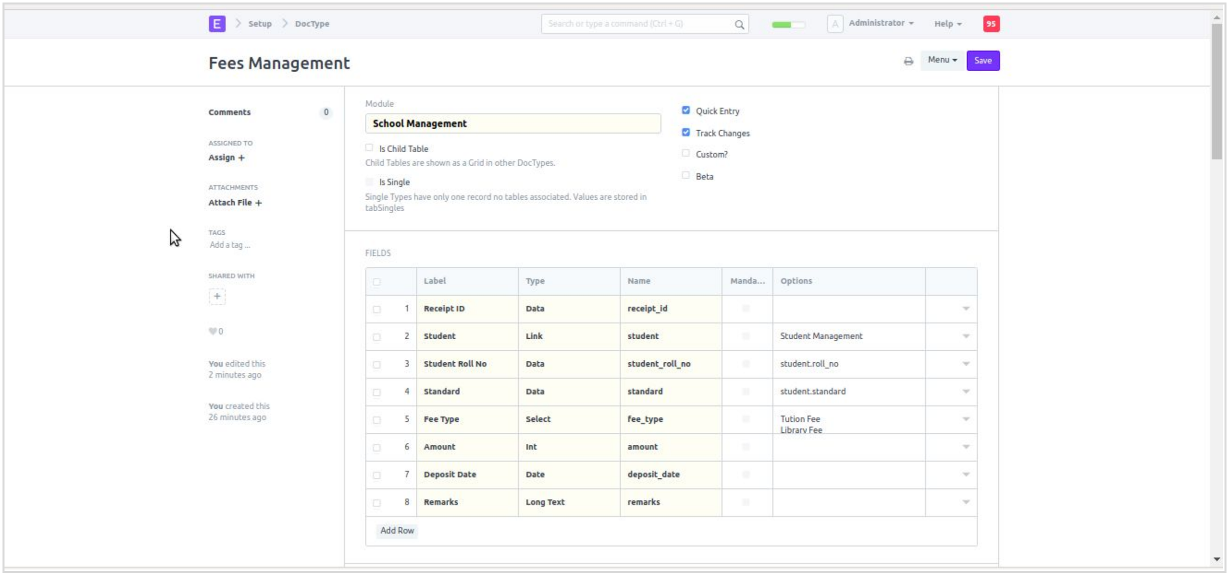
Module: School Management

- [ ] Is Child Table — Child Tables are shown as a Grid in other DocTypes.
- [ ] Is Single — Single Types have only one record no tables associated. Values are stored in tabSingles
- [✓] Quick Entry
- [✓] Track Changes
- [ ] Custom?
- [ ] Beta

FIELDS

	Label	Type	Name	Manda...	Options
1	Roll No	Data	roll_no		
2	Name	Data	name1		
3	Gender	Select	gender		Male Female
4	Date of Birth	Data	date_of_birth		
5	Address	Long Text	address		
6	State	Data	state		
7	Contact No	Int	contact_no		
8	Standard	Data	standard		
9	Status	Data	status		

Go to Developer menu > Navigate the documents link > Doctype
Click on “New” button and Create a Fee Management doctype



Fees Management

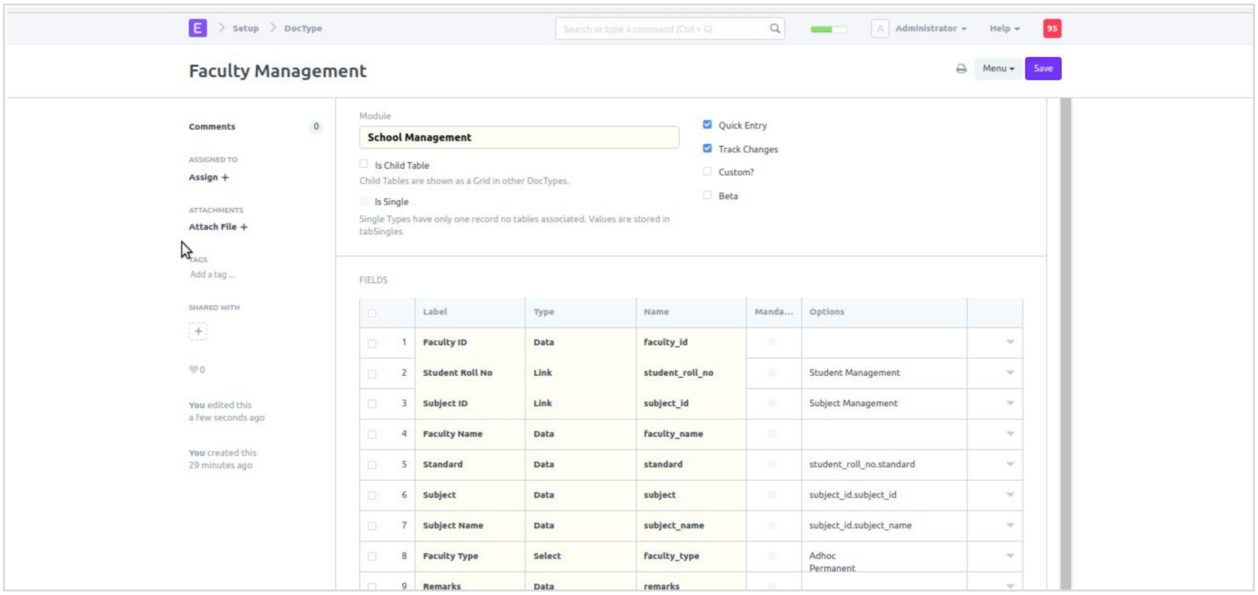
Module: School Management

- [ ] Is Child Table — Child Tables are shown as a Grid in other DocTypes.
- [ ] Is Single — Single Types have only one record no tables associated. Values are stored in tabSingles
- [✓] Quick Entry
- [✓] Track Changes
- [ ] Custom?
- [ ] Beta

FIELDS

	Label	Type	Name	Manda...	Options
1	Receipt ID	Data	receipt_id		
2	Student	Link	student		Student Management
3	Student Roll No	Data	student_roll_no		student.roll_no
4	Standard	Data	standard		student.standard
5	Fee Type	Select	fee_type		Tution Fee Library Fee
6	Amount	Int	amount		
7	Deposit Date	Date	deposit_date		
8	Remarks	Long Text	remarks		

Go to Developer menu > Navigate the documents link > Doctype
Click on “New” button and Create a Faculty Management doctype



Faculty Management

You edited this a few seconds ago

You created this 29 minutes ago

Module: School Management

☐ Is Child Table
Child Tables are shown as a Grid in other DocTypes.

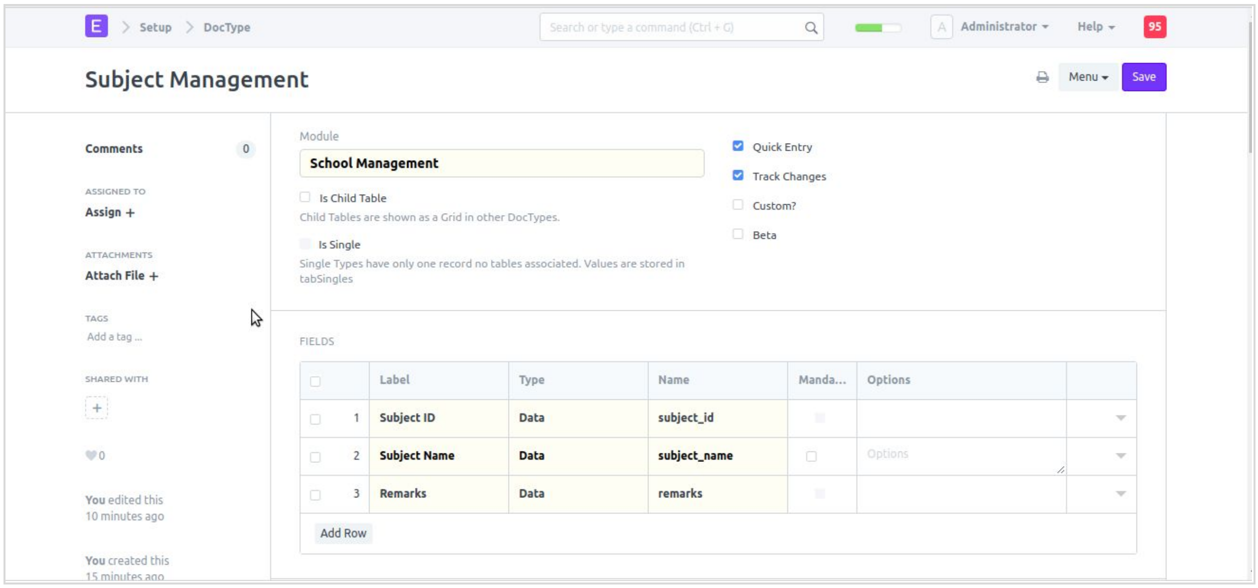
☐ Is Single
Single Types have only one record no tables associated. Values are stored in tabSingles

☑ Quick Entry
☑ Track Changes
☐ Custom?
☐ Beta

FIELDS

	Label	Type	Name	Manda...	Options
1	Faculty ID	Data	faculty_id	☐	
2	Student Roll No	Link	student_roll_no	☐	Student Management
3	Subject ID	Link	subject_id	☐	Subject Management
4	Faculty Name	Data	faculty_name	☐	
5	Standard	Data	standard	☐	student_roll_no.standard
6	Subject	Data	subject	☐	subject_id.subject_id
7	Subject Name	Data	subject_name	☐	subject_id.subject_name
8	Faculty Type	Select	faculty_type	☐	Adhoc Permanent
9	Remarks	Data	remarks	☐	

Go to Developer menu > Navigate the documents link > Doctype
Click on “New” button and Create a Subject Management doctype



Subject Management

You edited this 10 minutes ago

You created this 15 minutes ago

Module: School Management

☐ Is Child Table
Child Tables are shown as a Grid in other DocTypes.

☐ Is Single
Single Types have only one record no tables associated. Values are stored in tabSingles

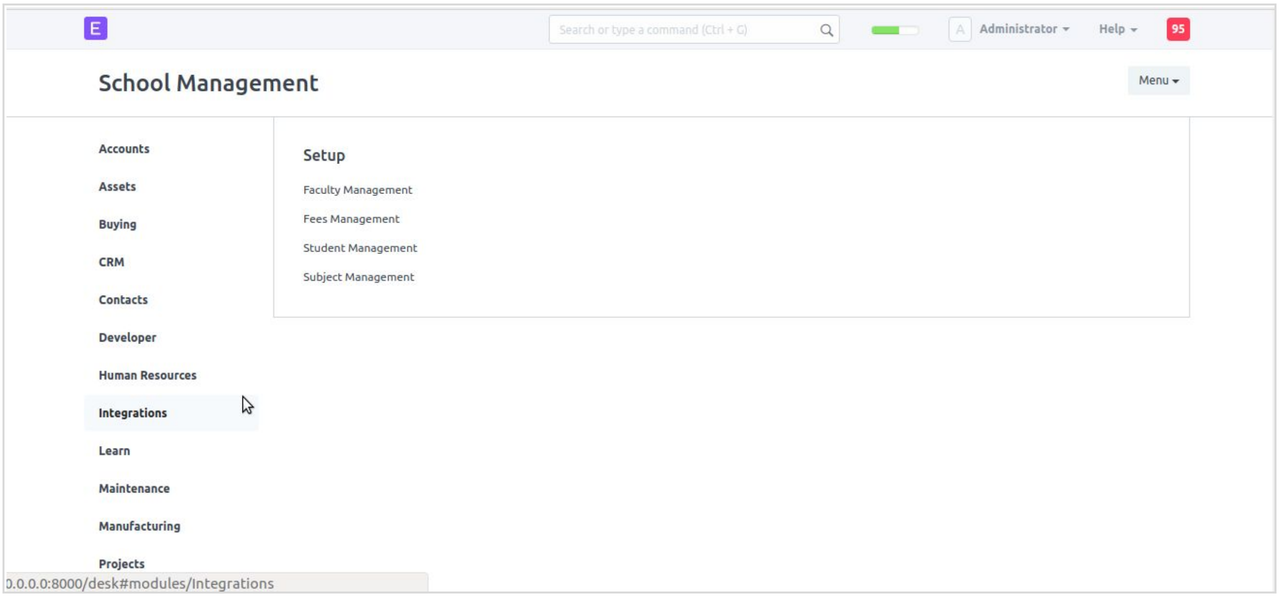
☑ Quick Entry
☑ Track Changes
☐ Custom?
☐ Beta

FIELDS

	Label	Type	Name	Manda...	Options
1	Subject ID	Data	subject_id	☐	
2	Subject Name	Data	subject_name	☐	Options
3	Remarks	Data	remarks	☐	

5. Access App

User can access docType or we can say our application from there main menu see the below screenshot.



School Management

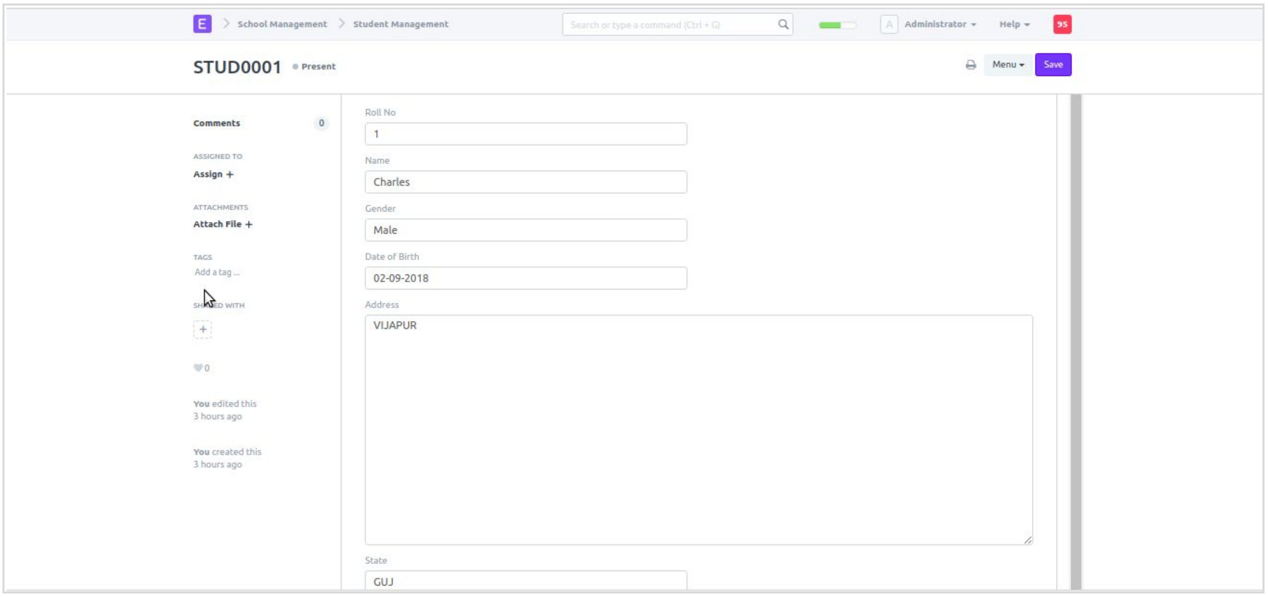
Accounts
Assets
Buying
CRM
Contacts
Developer
Human Resources
Integrations
Learn
Maintenance
Manufacturing
Projects

Setup

Faculty Management
Fees Management
Student Management
Subject Management

0.0.0.0:8000/desk#modules/Integrations

User can access student management form while click on Student Management link



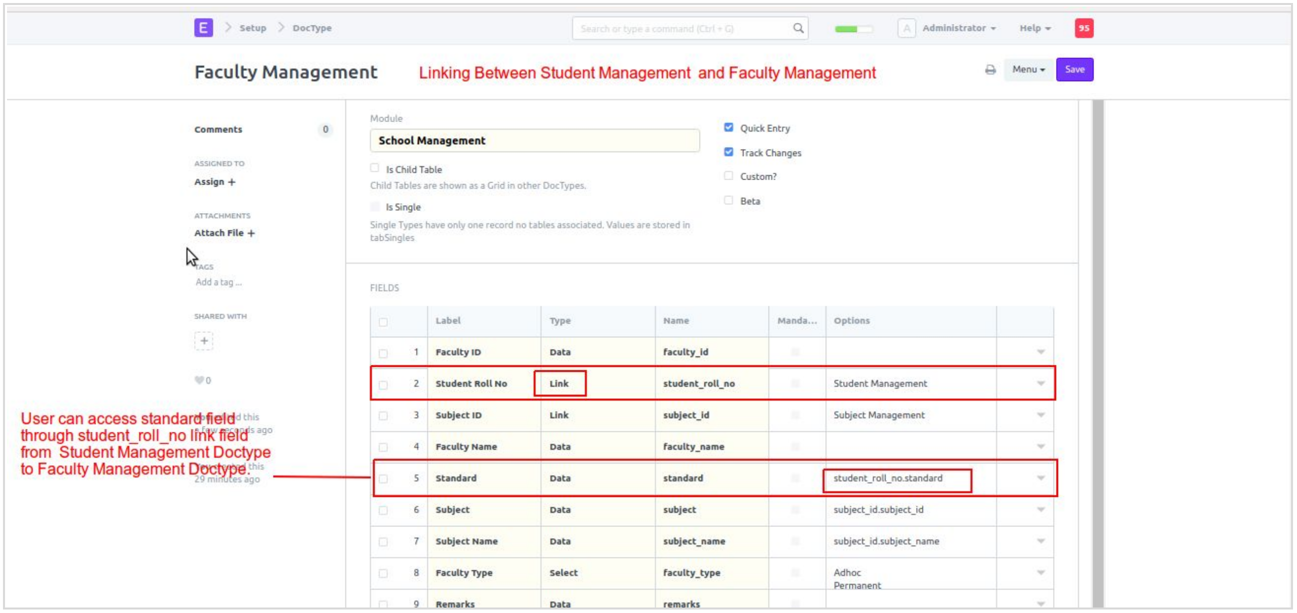
STUD0001 Present

You edited this 3 hours ago

You created this 3 hours ago

Roll No	1
Name	Charles
Gender	Male
Date of Birth	02-09-2018
Address	VIJAPUR
State	GUJ

6. Doctype Linking



Faculty Management

Linking Between Student Management and Faculty Management

User can access standard field through student_roll_no link field from Student Management Doctype to Faculty Management Doctype.

Module: School Management

- [✓] Quick Entry
- [✓] Track Changes
- [ ] Custom?
- [ ] Beta
- [ ] Is Child Table

Child Tables are shown as a Grid in other DocTypes.

- [ ] Is Single

Single Types have only one record no tables associated. Values are stored in tabSingles

FIELDS

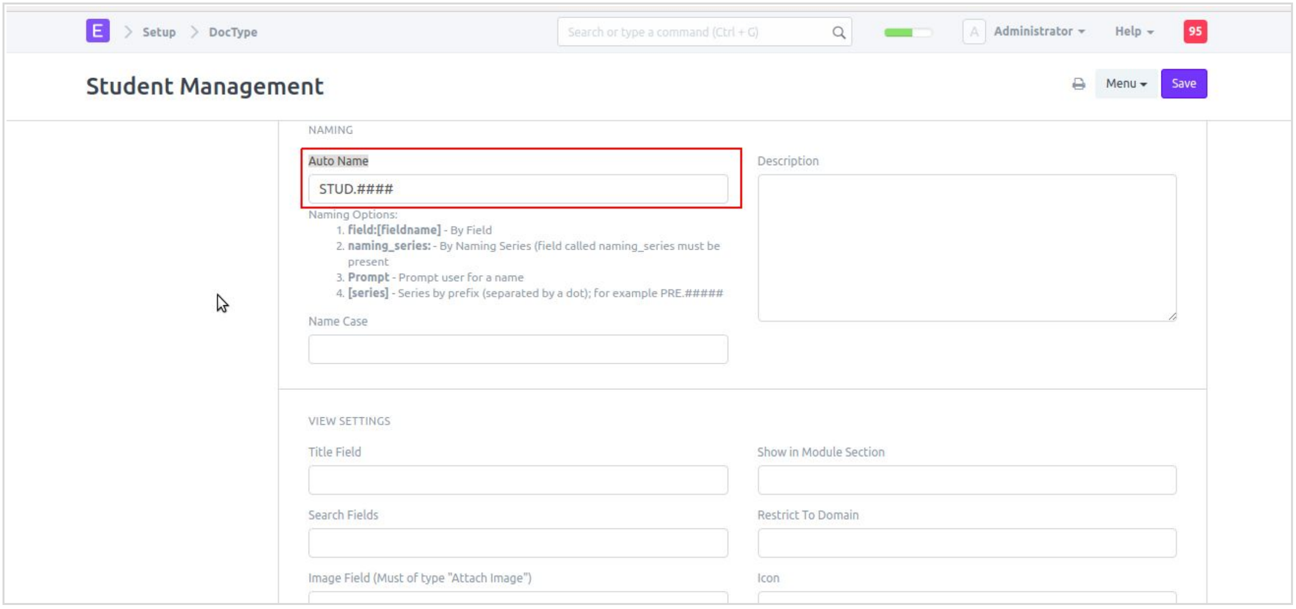
	Label	Type	Name	Manda...	Options
1	Faculty ID	Data	faculty_id		
2	Student Roll No	Link	student_roll_no		Student Management
3	Subject ID	Link	subject_id		Subject Management
4	Faculty Name	Data	faculty_name		
5	Standard	Data	standard		student_roll_no.standard
6	Subject	Data	subject		subject_id.subject_id
7	Subject Name	Data	subject_name		subject_id.subject_name
8	Faculty Type	Select	faculty_type		Adhoc Permanent
9	Remarks	Data	remarks		

7. Doctype Naming

DocTypes named defined in four ways:

1. Series
2. Field
3. By controller (code)
4. Prompt

1. Series: : Series by prefix (Separate by dot)
For example: STUD.####
User can set Naming Series in Auto Name field:



Student Management

NAMING

Auto Name
STUD.####

Naming Options:
1. **field:[fieldname]** - By Field
2. **naming_series:** - By Naming Series (field called naming_series must be present
3. **Prompt** - Prompt user for a name
4. **[series]** - Series by prefix (separated by a dot); for example PRE.#####

Name Case

Description

VIEW SETTINGS

Title Field

Show in Module Section

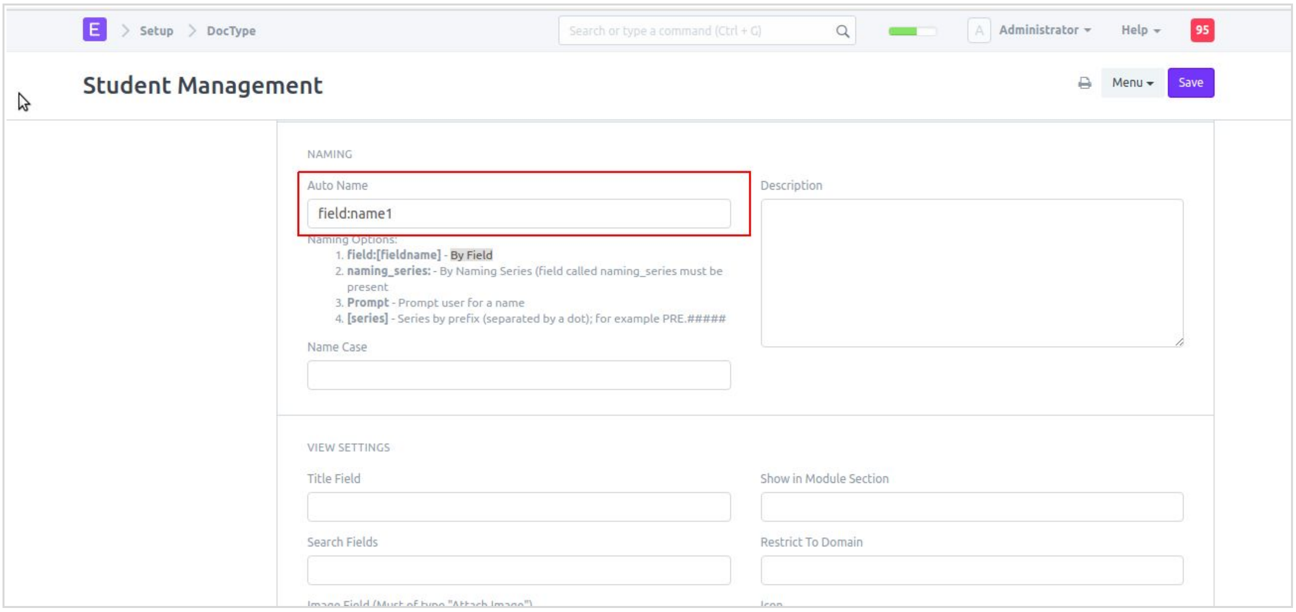
Search Fields

Restrict To Domain

Image Field (Must of type "Attach Image")

Icon

2. Series: : Series by prefix (Separate by dot)
Syntax: field:<<field name>>
For example: field:name1
User can set Naming Series in Auto Name field:



Student Management

NAMING

Auto Name
field:name1

Naming Options:
1. **field:[fieldname]** - By Field
2. **naming_series:** - By Naming Series (field called naming_series must be present
3. **Prompt** - Prompt user for a name
4. **[series]** - Series by prefix (separated by a dot); for example PRE.#####

Name Case

Description

VIEW SETTINGS

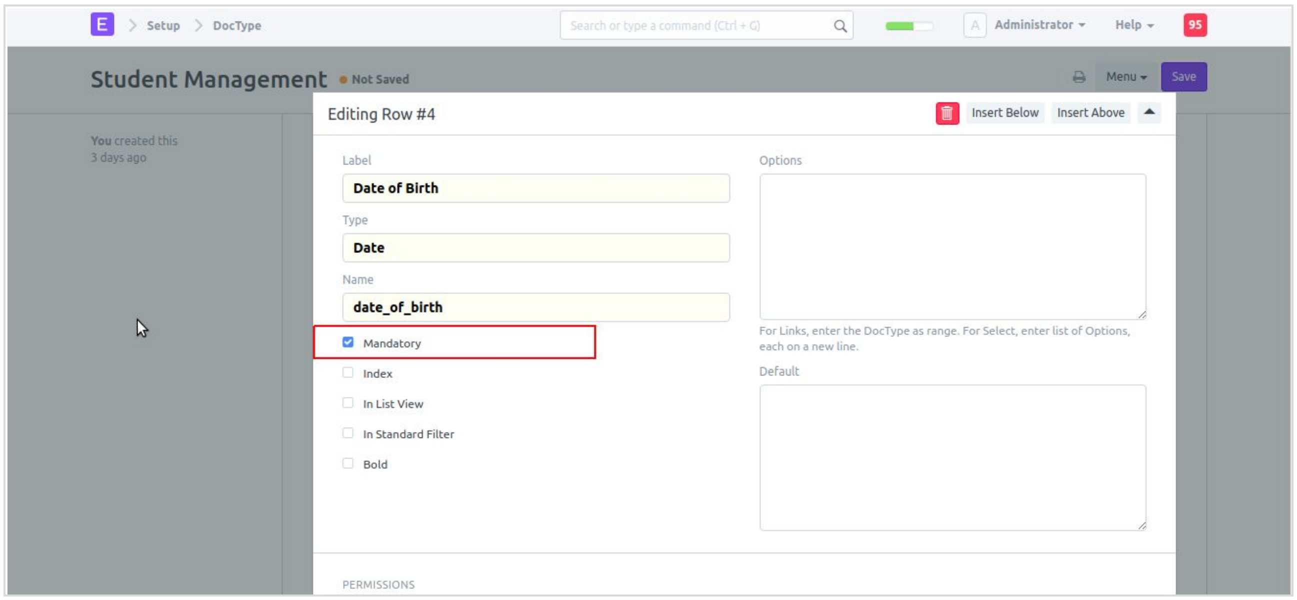
Title Field

Show in Module Section

Search Fields

Restrict To Domain

8. Set Mandatory Field



Setup > DocType

Search or type a command (Ctrl + G)

Administrator | Help | 95

Student Management Not Saved

Menu | Save

You created this 3 days ago

Editing Row #4

Insert Below | Insert Above

Label: Date of Birth

Type: Date

Name: date_of_birth

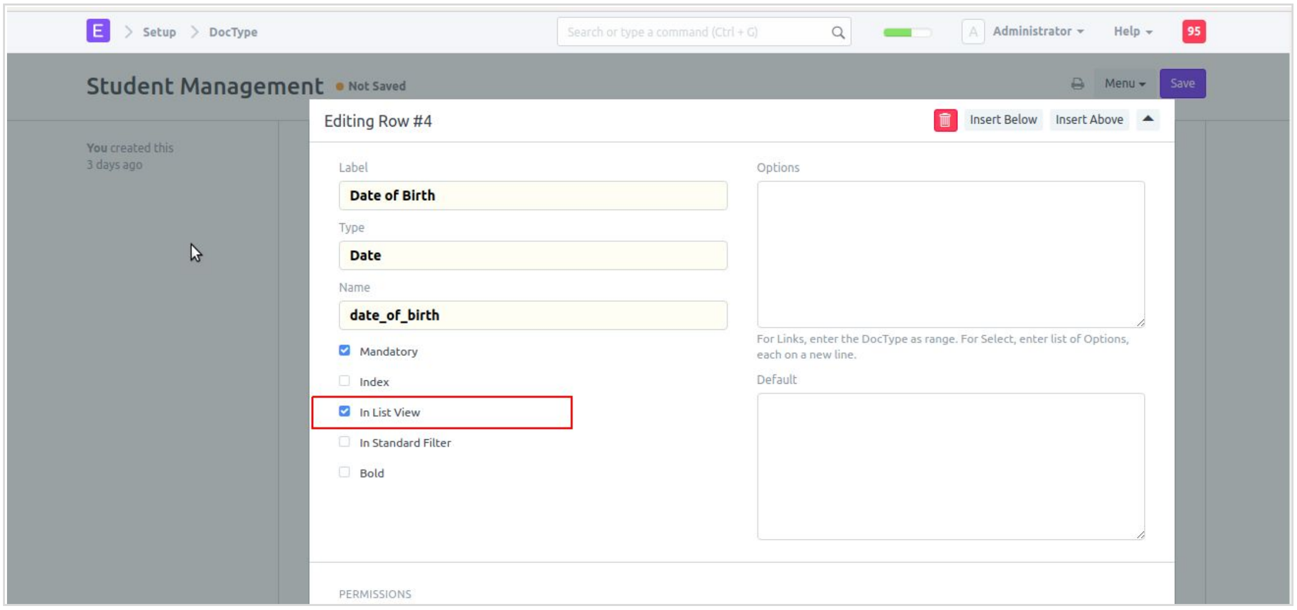
- [x] Mandatory
- [ ] Index
- [ ] In List View
- [ ] In Standard Filter
- [ ] Bold

Options

For Links, enter the DocType as range. For Select, enter list of Options, each on a new line.

Default

PERMISSIONS



Setup > DocType

Search or type a command (Ctrl + G)

Administrator | Help | 95

Student Management Not Saved

Menu | Save

You created this 3 days ago

Editing Row #4

Insert Below | Insert Above

Label: Date of Birth

Type: Date

Name: date_of_birth

- [x] Mandatory
- [ ] Index
- [x] In List View
- [ ] In Standard Filter
- [ ] Bold

Options

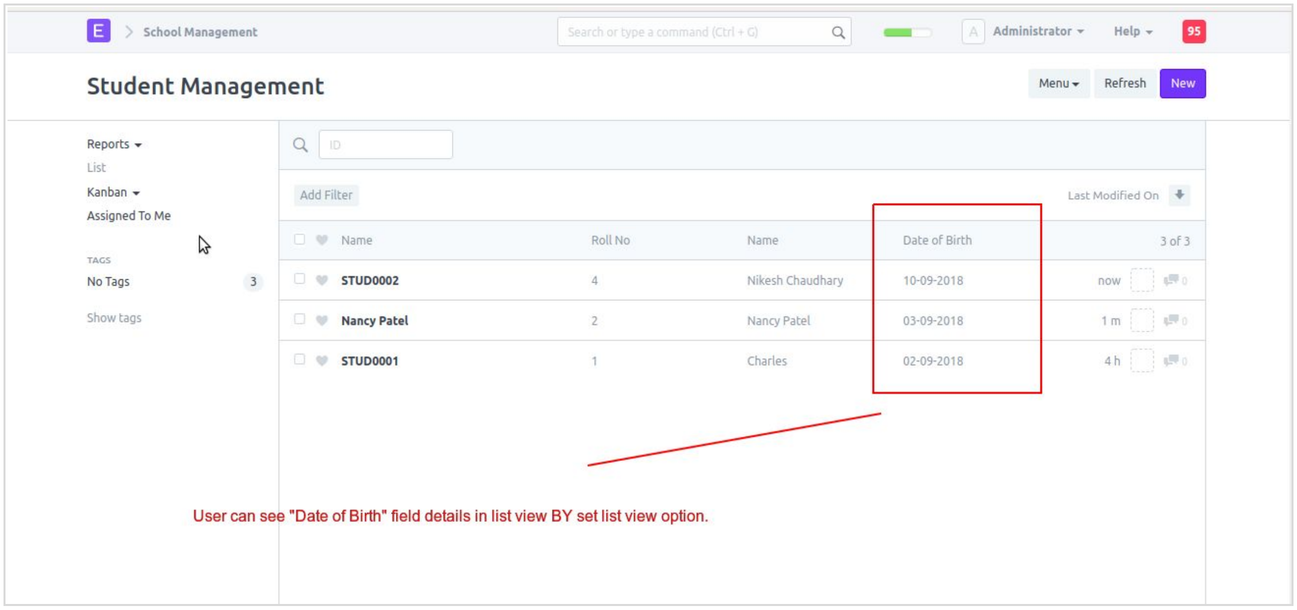
For Links, enter the DocType as range. For Select, enter list of Options, each on a new line.

Default

PERMISSIONS

9. Set List View Option

Field detail is shown in list view. See the below screenshot:

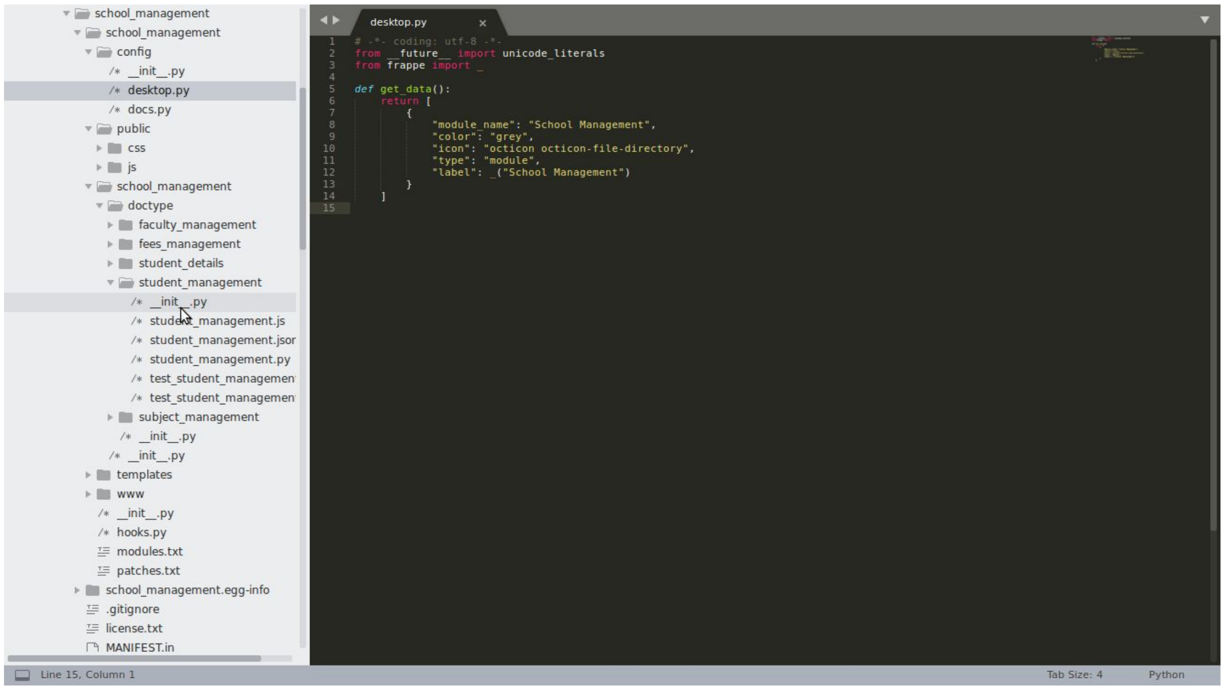


	Name	Roll No	Name	Date of Birth	3 of 3
	STUD0002	4	Nikesh Chaudhary	10-09-2018	now 0
	Nancy Patel	2	Nancy Patel	03-09-2018	1 m 0
	STUD0001	1	Charles	02-09-2018	4 h 0

User can see "Date of Birth" field details in list view BY set list view option.

10. App Directory Structure

The application will be created in a folder called school_managementand will have the following structure:



```
school_management
  school_management
    config
      __init__.py
      desktop.py
      docs.py
    public
      css
      js
    school_management
      doctype
        faculty_management
        fees_management
        student_details
        student_management
          __init__.py
          student_management.js
          student_management.jsor
          student_management.py
          test_student_managemen
          test_student_managemen
        subject_management
        __init__.py
      __init__.py
    templates
    www
    __init__.py
    hooks.py
    modules.txt
    patches.txt
  school_management.egg-info
  .gitignore
  license.txt
  MANIFEST.in
```

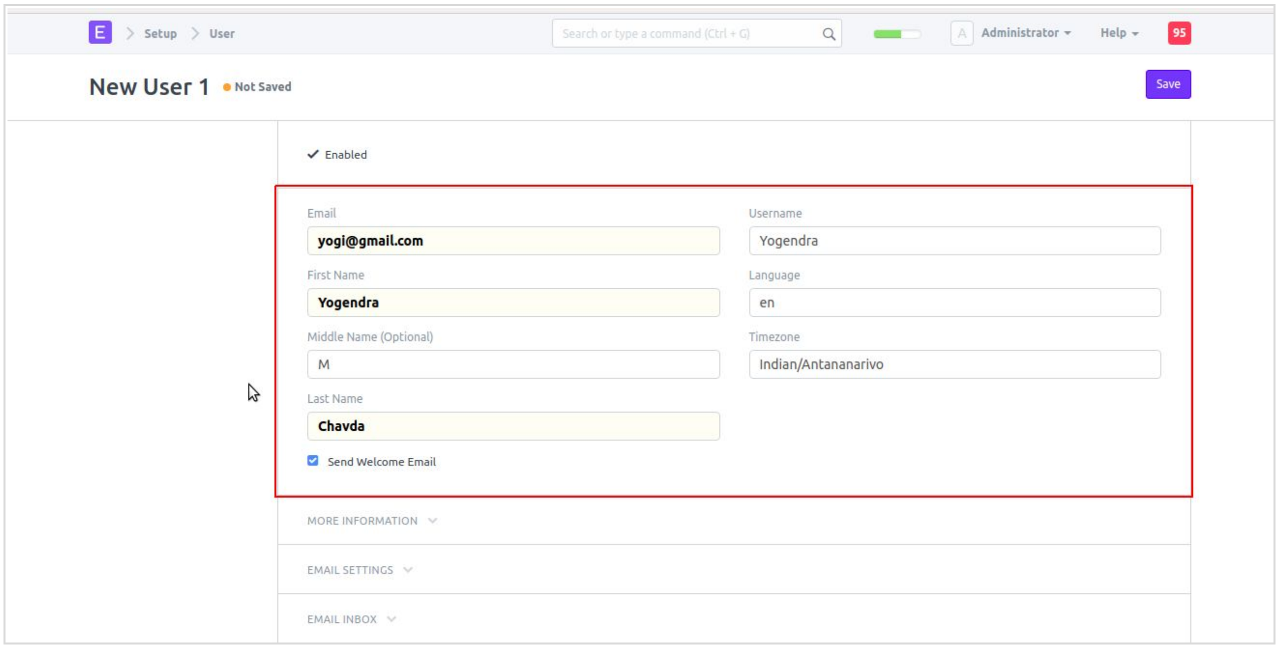
desktop.py

```python
# -*- coding: utf-8 -*-
from __future__ import unicode_literals
from frappe import _

def get_data():
    return [
        {
            "module_name": "School Management",
            "color": "grey",
            "icon": "octicon octicon-file-directory",
            "type": "module",
            "label": _("School Management")
        }
    ]
```

- ❑ Config folder contains application configuration info
- ❑ Desktop.py is where desktop icons can be added to the Desk
- ❑ hooks.py is where integrations with the environment and other applications is mentioned.
- ❑ library_management (inner) is a module that is bootstrapped. In Frappe, a module is where model and controller files reside.
- ❑ modules.txt contains list of modules in the app. When you create a new module, it is required that you update it in this file.
- ❑ patches.txt is where migration patches are written. They are python module references using the dot notation.
- ❑ Templates is the folder where webview templates are maintained. Templates for Login and other standard pages are bootstrapped in frappe.
- ❑ Generators are where templates for models are maintained, where each model instance has a separate web route, for example a Blog Post . where each post has its unique web url. In Frappe, the templating engine used is Jinja2
- ❑ Pages is where single route templates are maintained. For example for a "/blog" type of page.

11. Creating Users

We can create user from the menu setup > User > Click on new button



New User 1 • Not Saved

✓ Enabled

Email: yogi@gmail.com	Username: Yogendra
First Name: Yogendra	Language: en
Middle Name (Optional): M	Timezone: Indian/Antananarivo
Last Name: Chavda	

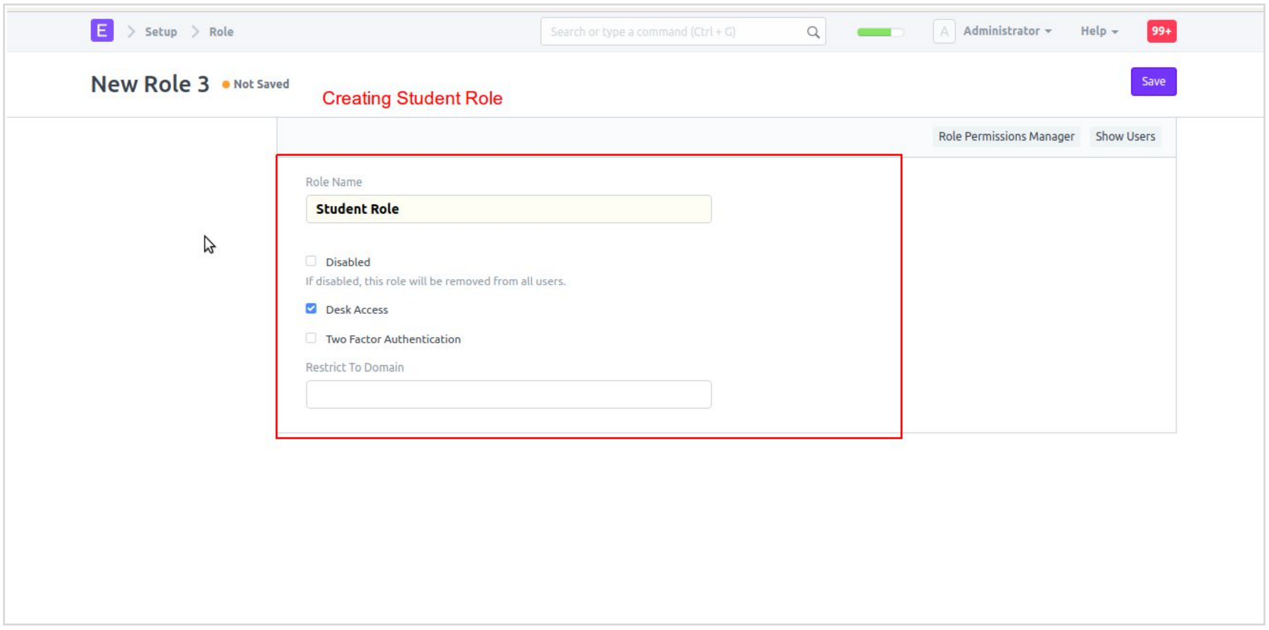
☑ Send Welcome Email

12. Creating Roles

Go to setup > Role > Click on new button

Before creating Models, we must create Roles so that we can set permissions on the Model. There are two Roles we will create:

1. Student Role
2. Faculty Role



New Role 3 ● Not Saved

Role Name: Student Role

☐ Disabled
If disabled, this role will be removed from all users.

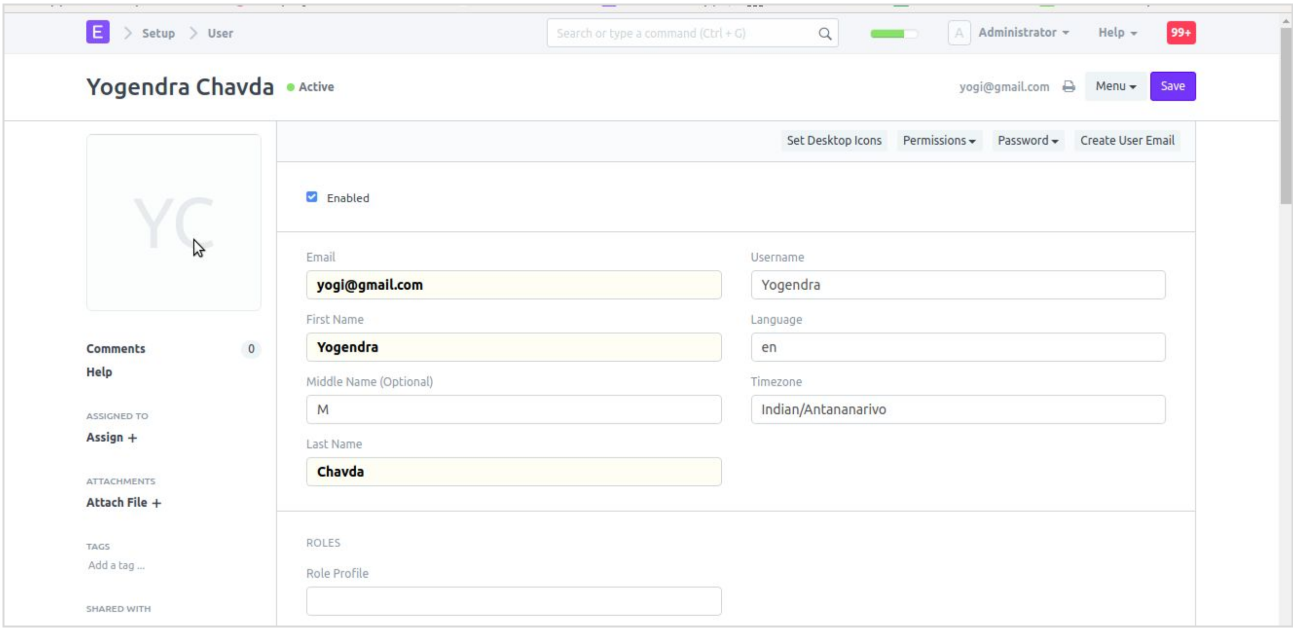
☑ Desk Access

☐ Two Factor Authentication

Restrict To Domain

13. Setting Roles : Go to setup > User

We can allocate role for particular user by check the role. See the below screenshot.



Setup > User

Search or type a command (Ctrl + G)

Administrator | Help | 99+

Yogendra Chavda ● Active

yogi@gmail.com | Menu | Save

Set Desktop Icons | Permissions | Password | Create User Email

[✓] Enabled

Email: yogi@gmail.com	Username: Yogendra
First Name: Yogendra	Language: en
Middle Name (Optional): M	Timezone: Indian/Antananarivo
Last Name: Chavda	

Comments 0

Help

ASSIGNED TO

Assign +

ATTACHMENTS

Attach File +

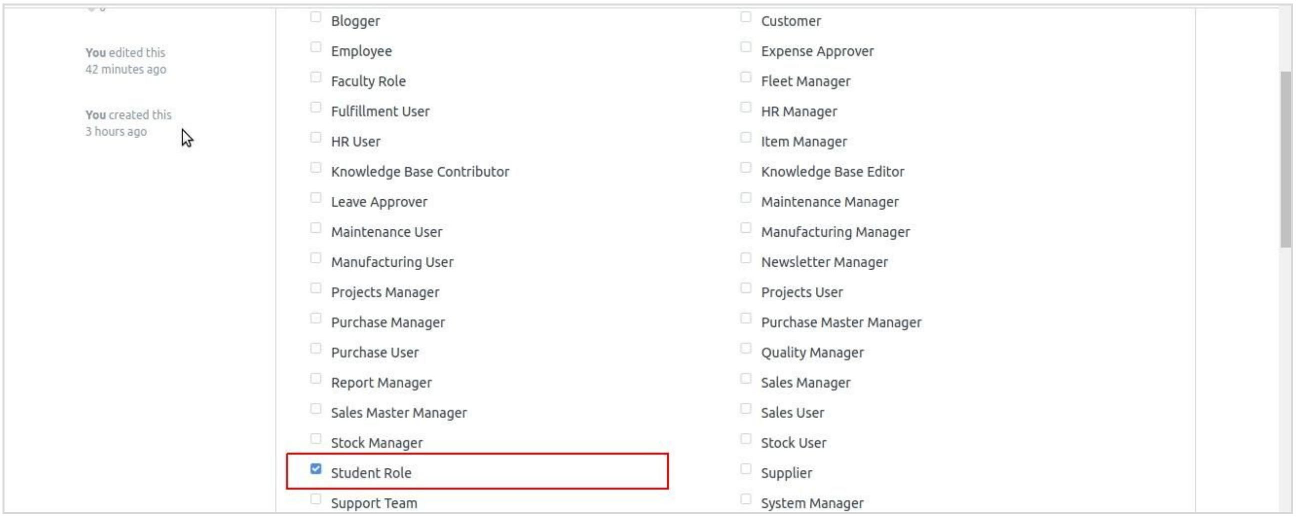
TAGS

Add a tag ...

SHARED WITH

ROLES

Role Profile



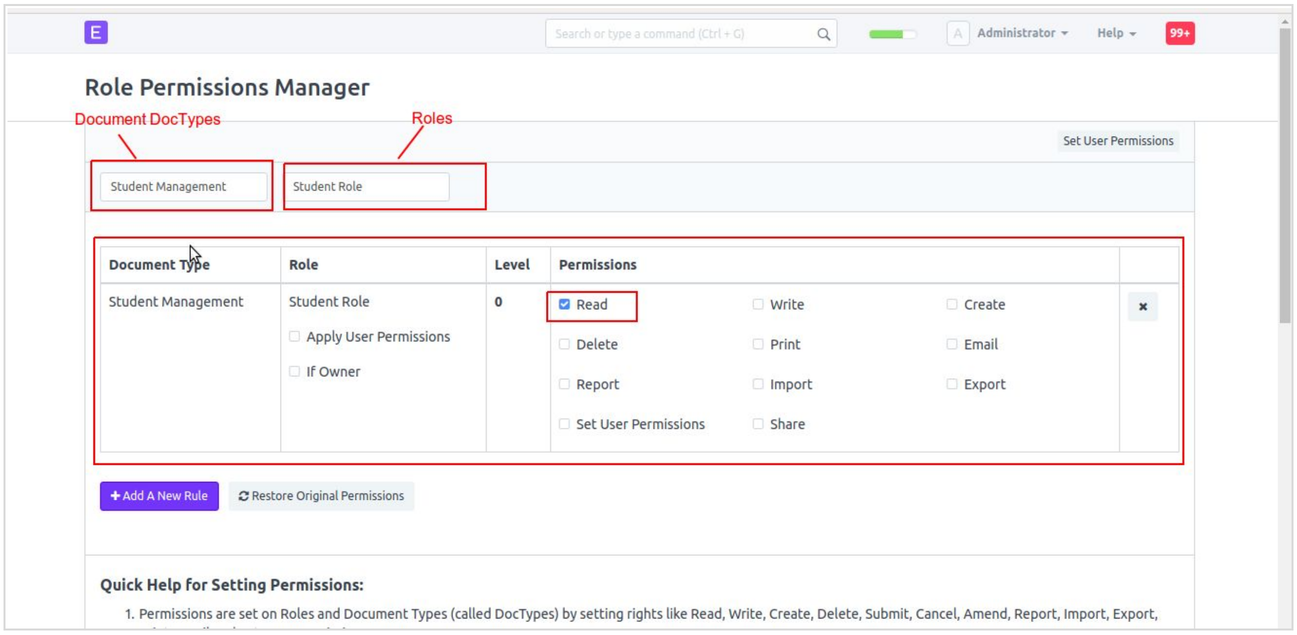
You edited this 42 minutes ago

You created this 3 hours ago

☐ Blogger	☐ Customer
☐ Employee	☐ Expense Approver
☐ Faculty Role	☐ Fleet Manager
☐ Fulfillment User	☐ HR Manager
☐ HR User	☐ Item Manager
☐ Knowledge Base Contributor	☐ Knowledge Base Editor
☐ Leave Approver	☐ Maintenance Manager
☐ Maintenance User	☐ Manufacturing Manager
☐ Manufacturing User	☐ Newsletter Manager
☐ Projects Manager	☐ Projects User
☐ Purchase Manager	☐ Purchase Master Manager
☐ Purchase User	☐ Quality Manager
☐ Report Manager	☐ Sales Manager
☐ Sales Master Manager	☐ Sales User
☐ Stock Manager	☐ Stock User
✓ Student Role	☐ Supplier
☐ Support Team	☐ System Manager

14. Permissions

We can apply permissions for a specific role using “Role Permissions Manager”.
Goto setup > Role Permissions Manager



Role Permissions Manager

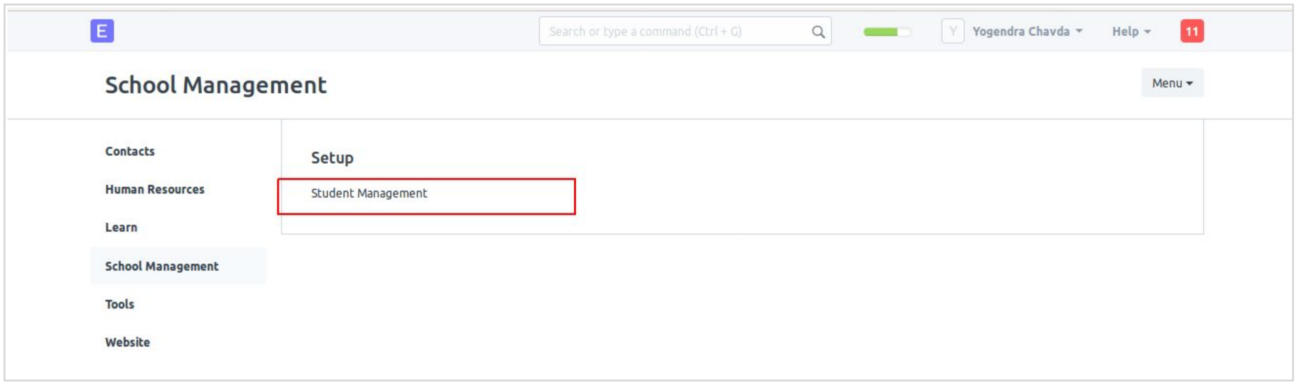
Student Management | Student Role

Document Type	Role	Level	Permissions			
Student Management	Student Role ☐ Apply User Permissions ☐ If Owner	0	☑ Read ☐ Delete ☐ Report ☐ Set User Permissions	☐ Write ☐ Print ☐ Import ☐ Share	☐ Create ☐ Email ☐ Export	×

Quick Help for Setting Permissions:

1. Permissions are set on Roles and Document Types (called DocTypes) by setting rights like Read, Write, Create, Delete, Submit, Cancel, Amend, Report, Import, Export,

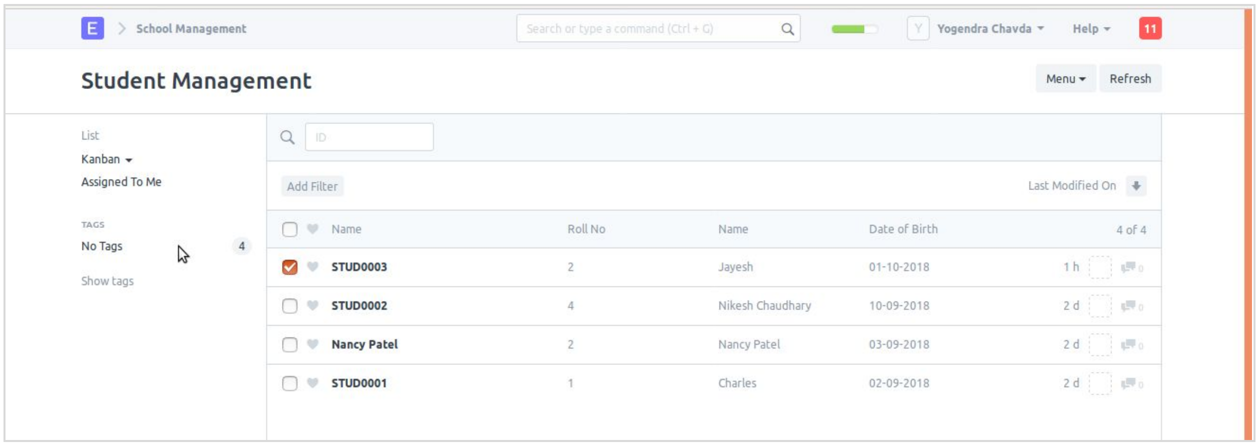
After applied permissions, User can access the application through desk.



School Management

Setup

Student Management

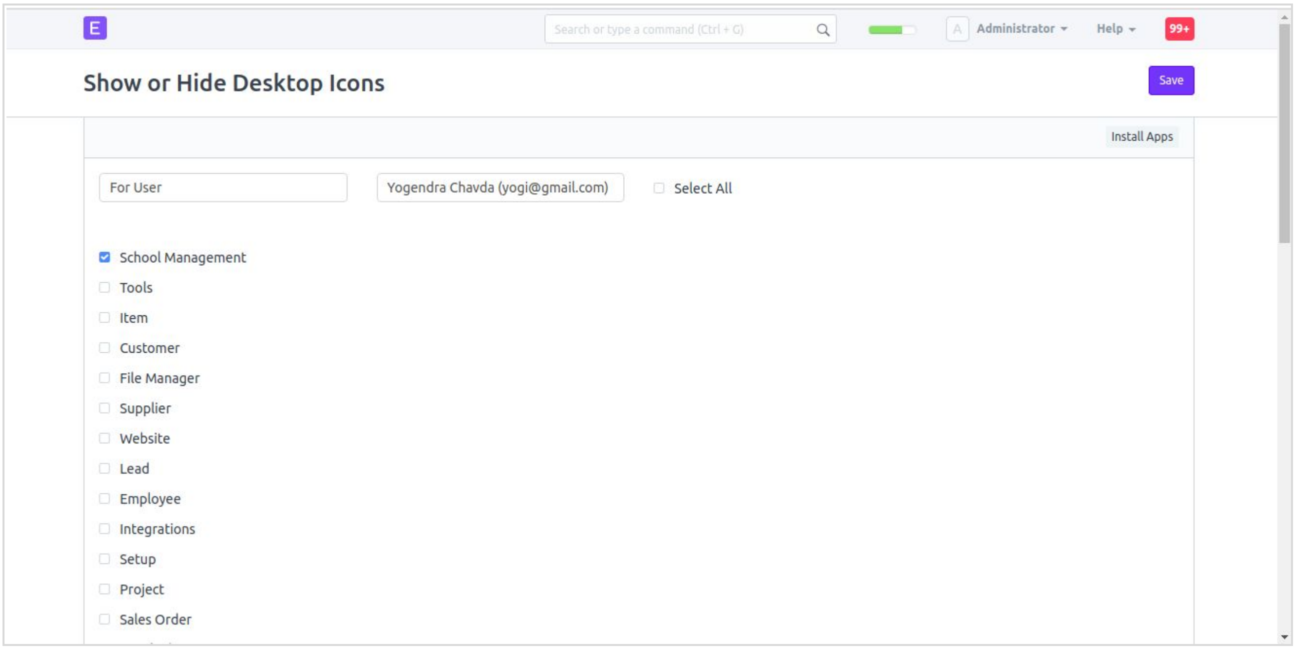


Student Management

Name	Roll No	Name	Date of Birth	4 of 4
STUD0003	2	Jayesh	01-10-2018	1 h
STUD0002	4	Nikesh Chaudhary	10-09-2018	2 d
Nancy Patel	2	Nancy Patel	03-09-2018	2 d
STUD0001	1	Charles	02-09-2018	2 d

15. Show or Hide Desktop Icon

User can show or hide module icon by check/uncheck option.
Go to > Setup > Show/Hide Modules



Show or Hide Desktop Icons

For User: Yogendra Chavda (yogi@gmail.com)

- [ ] Select All

- [x] School Management
- [ ] Tools
- [ ] Item
- [ ] Customer
- [ ] File Manager
- [ ] Supplier
- [ ] Website
- [ ] Lead
- [ ] Employee
- [ ] Integrations
- [ ] Setup
- [ ] Project
- [ ] Sales Order

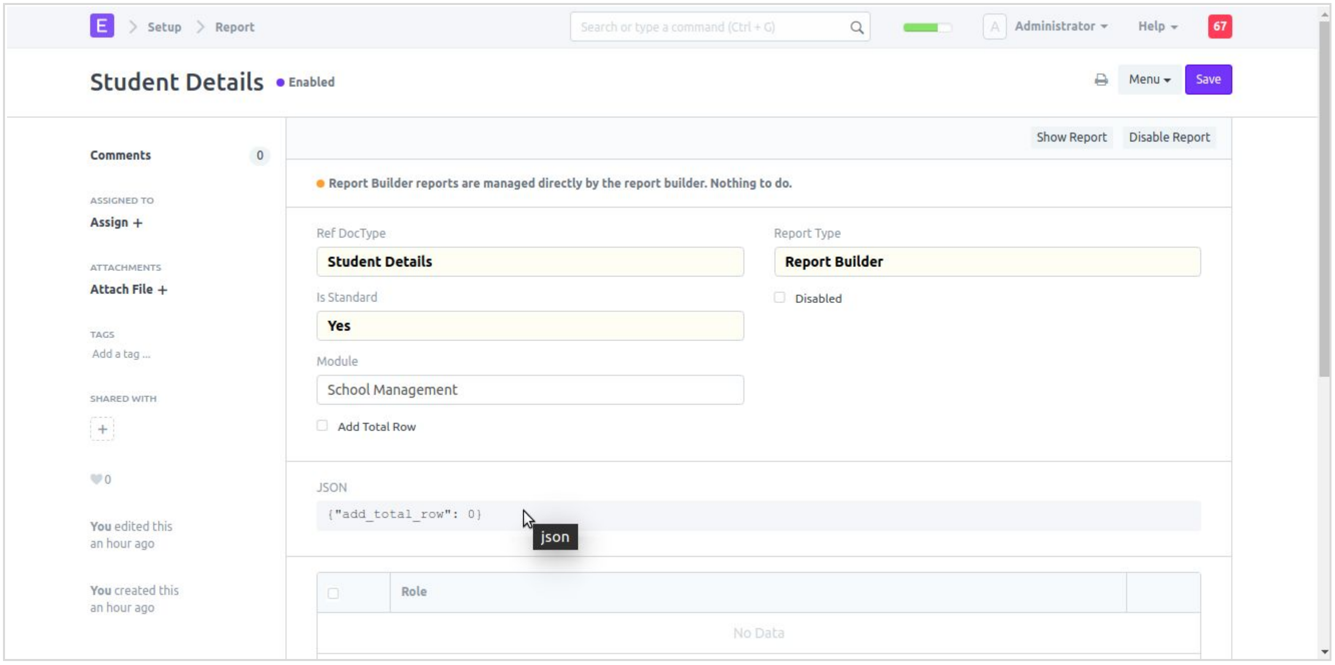
16. Create a Report

1. Analysis Report

User can create analysis report through below menu:
Go to > Developer > Report menu

Fill up the report details :

Report Name: Student Details,
Report Type: Report Builder,
Ref DocType: Student Details,
Is Standard: Yes,
Module: School Management



Student Details ● Enabled

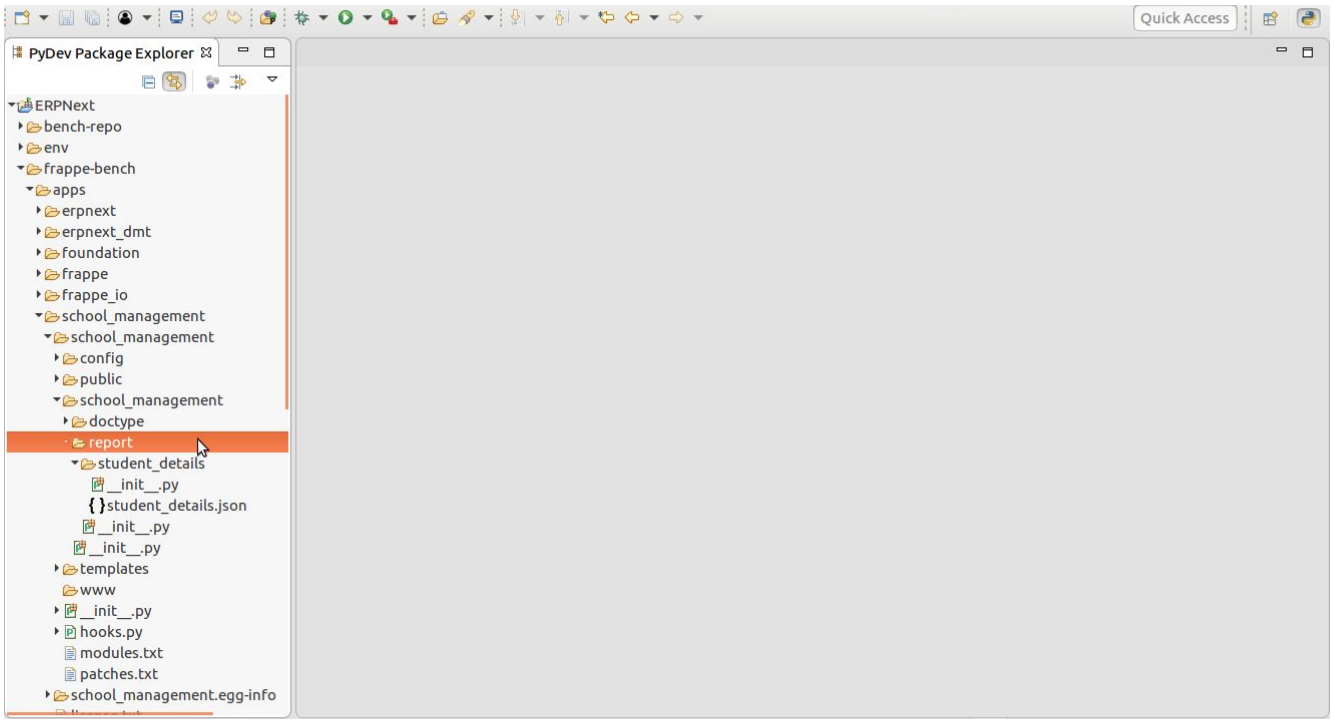
● Report Builder reports are managed directly by the report builder. Nothing to do.

Ref DocType: Student Details	Report Type: Report Builder
Is Standard: Yes	☐ Disabled
Module: School Management	
☐ Add Total Row	

JSON
```
{"add_total_row": 0}
```

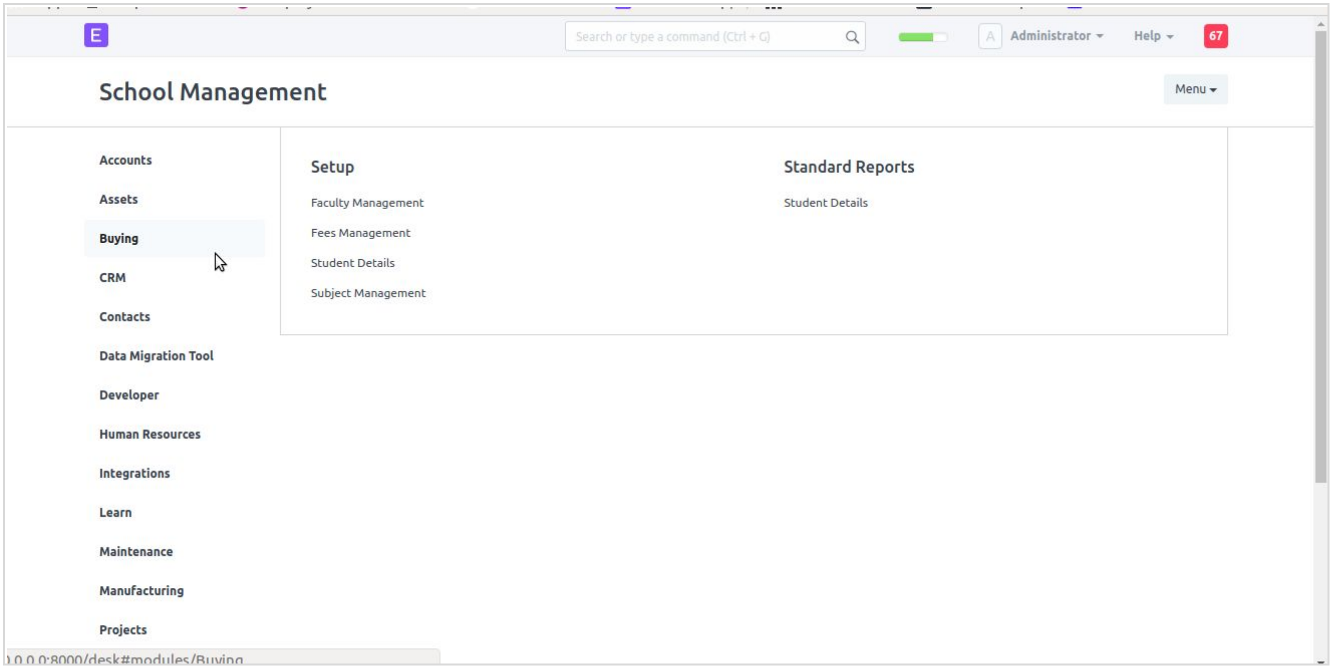
☐	Role	
	No Data	

Report folder automatically created under the custom app

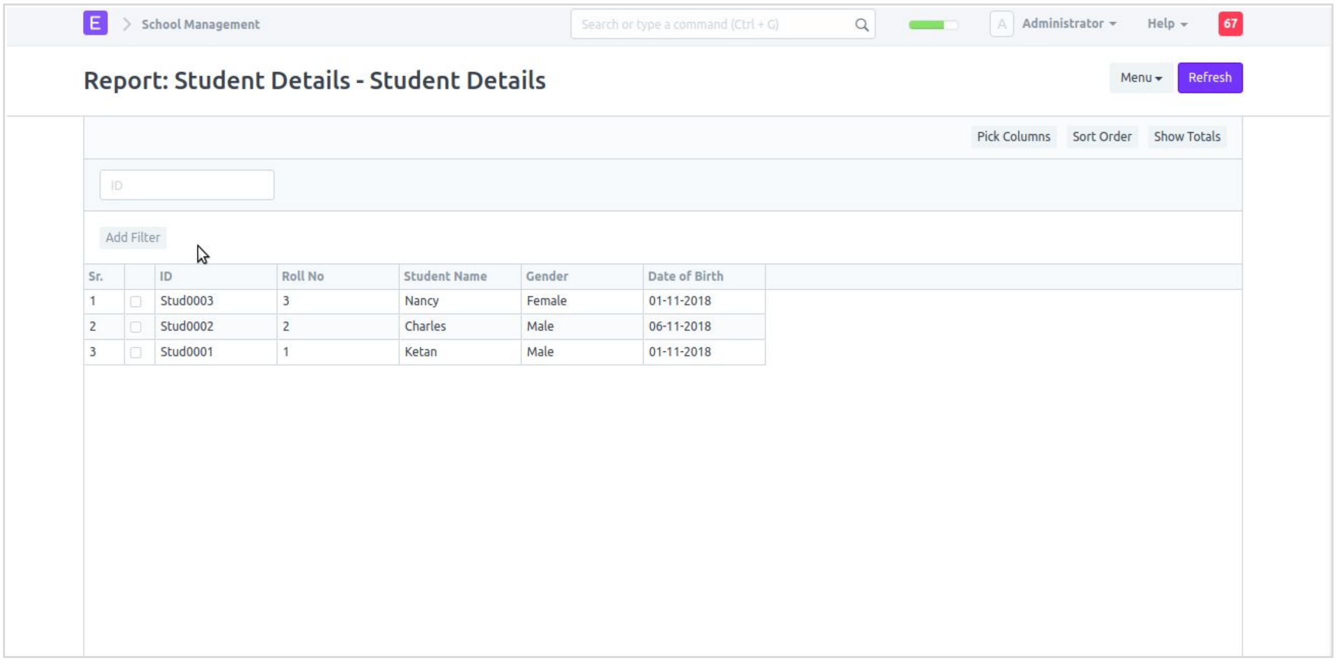


```
PyDev Package Explorer
▾ERPNext
  ▸bench-repo
  ▸env
  ▾frappe-bench
    ▾apps
      ▸erpnext
      ▸erpnext_dmt
      ▸foundation
      ▸frappe
      ▸frappe_io
      ▾school_management
        ▾school_management
          ▸config
          ▸public
          ▾school_management
            ▸doctype
            ▾report
              ▾student_details
                  __init__.py
                  student_details.json
                __init__.py
              __init__.py
          ▸templates
           www
          ▸__init__.py
          ▸hooks.py
           modules.txt
           patches.txt
        ▸school_management.egg-info
```

User can access student Details report under the Standard Reports section



Report View

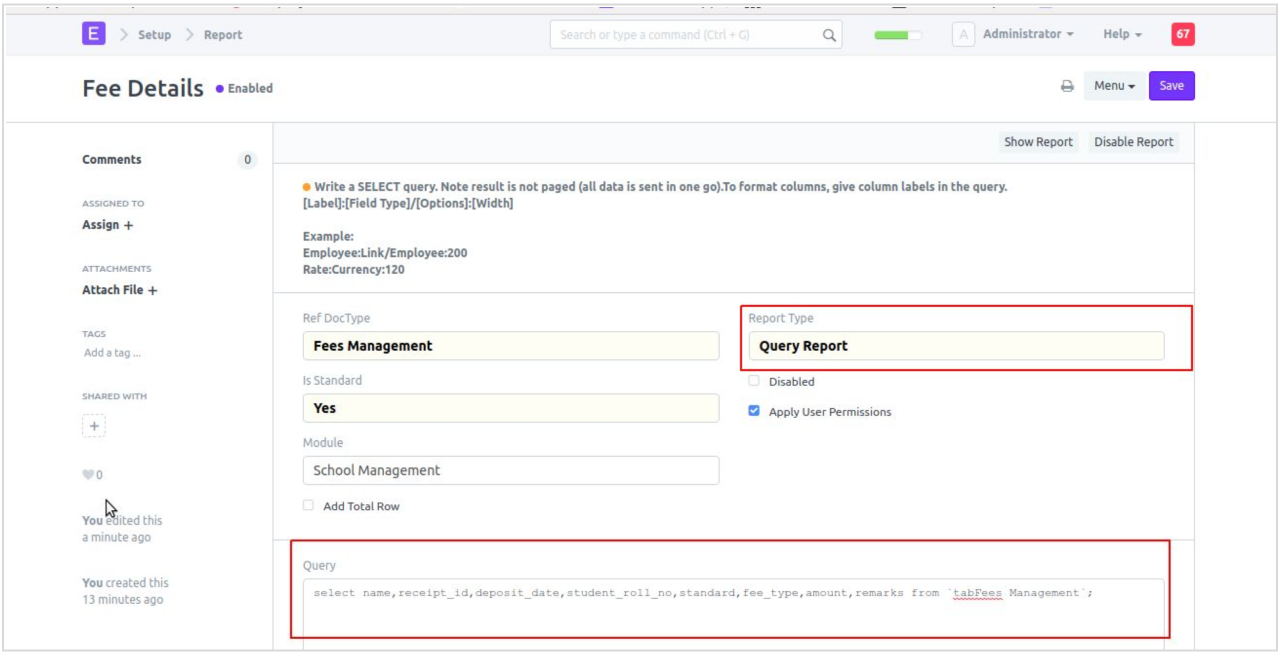


Report: Student Details - Student Details

Sr.		ID	Roll No	Student Name	Gender	Date of Birth
1	☐	Stud0003	3	Nancy	Female	01-11-2018
2	☐	Stud0002	2	Charles	Male	06-11-2018
3	☐	Stud0001	1	Ketan	Male	01-11-2018

2. Query Report
User can create Query Report through below menu:
Go to > Developer > Report menu

Fill up the report details :
Report Name: Fee Details,
Report Type: Query Report,
Ref DocType: Fees Management,
Is Standard: Yes,
Module: School Management



Fee Details • Enabled

Write a SELECT query. Note result is not paged (all data is sent in one go).To format columns, give column labels in the query.
[Label]:[Field Type]/[Options]:[Width]

Example:
Employee:Link/Employee:200
Rate:Currency:120

Ref DocType: Fees Management

Report Type: Query Report

Is Standard: Yes

Disabled: ☐

Apply User Permissions: ✓

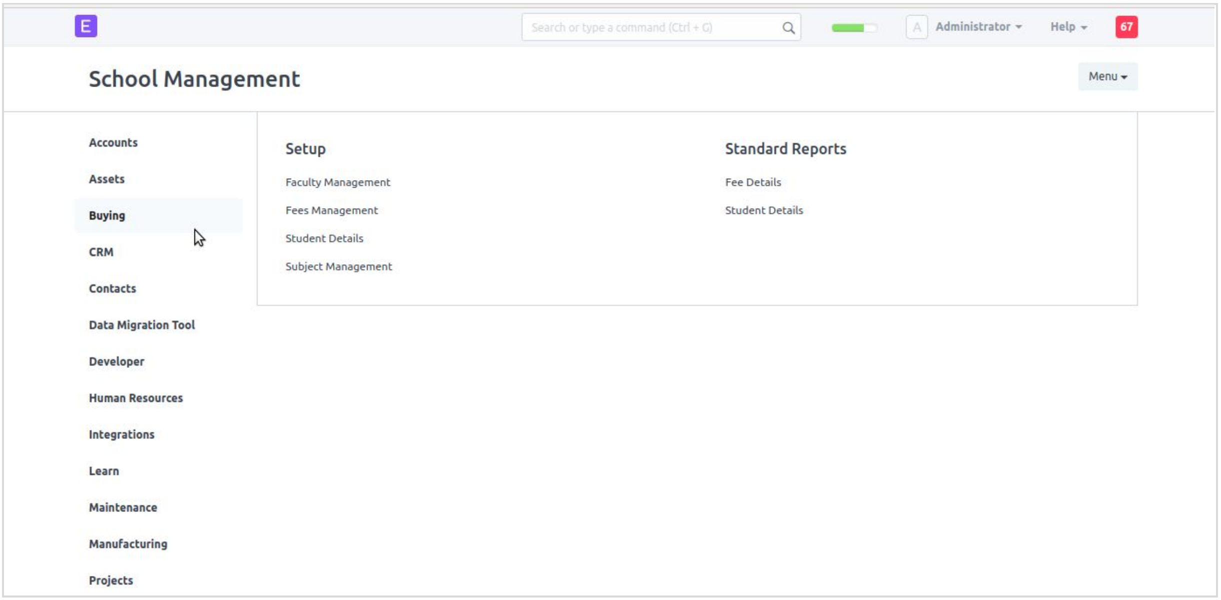
Module: School Management

Add Total Row: ☐

Query

```
select name,receipt_id,deposit_date,student_roll_no,standard,fee_type,amount,remarks from `tabFees Management`;
```

User can access Fee Details report under the Standard Reports section

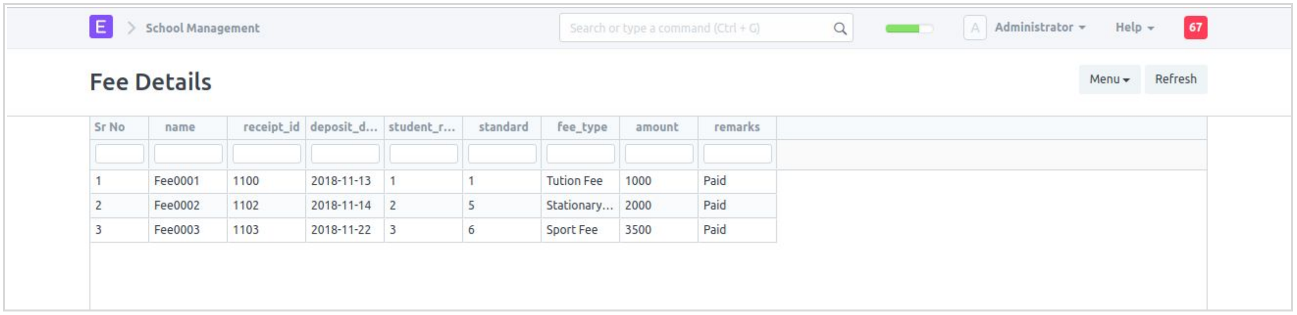


School Management

Setup
- Faculty Management
- Fees Management
- Student Details
- Subject Management

Standard Reports
- Fee Details
- Student Details

Report View

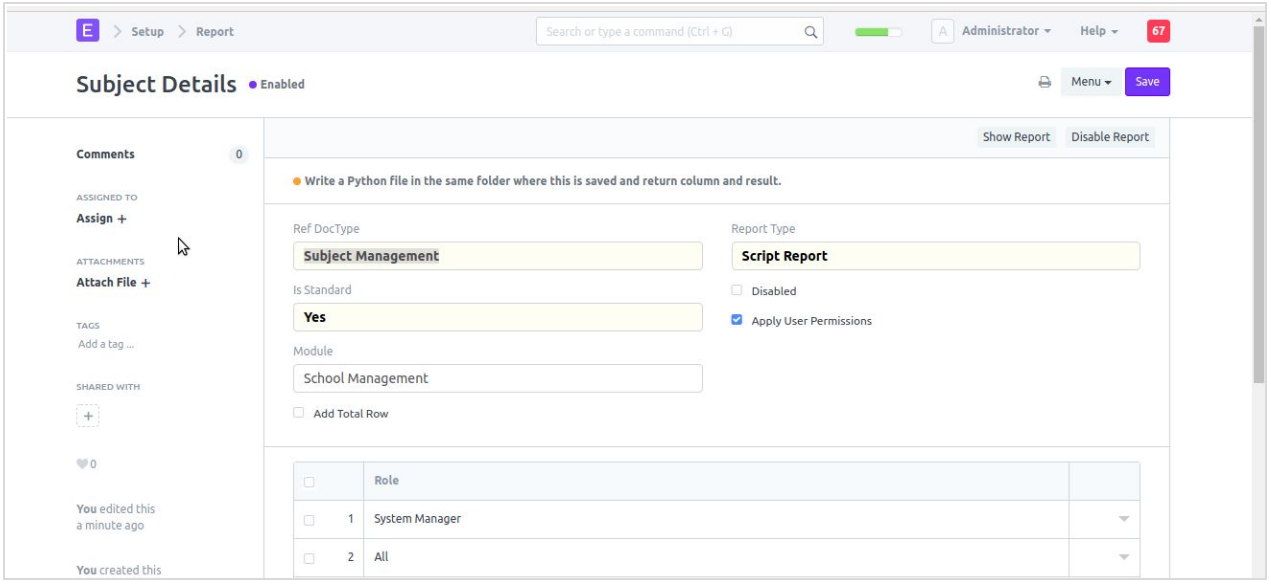

School Management

Fee Details

Sr No	name	receipt_id	deposit_d...	student_r...	standard	fee_type	amount	remarks
1	Fee0001	1100	2018-11-13	1	1	Tution Fee	1000	Paid
2	Fee0002	1102	2018-11-14	2	5	Stationary...	2000	Paid
3	Fee0003	1103	2018-11-22	3	6	Sport Fee	3500	Paid

3. Script Report
User can create Script Report through below menu:
Go to > Developer > Report menu

Fill up report details :
Report Name: Subject Details,
Report Type: Script Report,
Ref DocType: Subject Management,
Is Standard: Yes,
Module: School Management


Setup > Report

Subject Details • Enabled

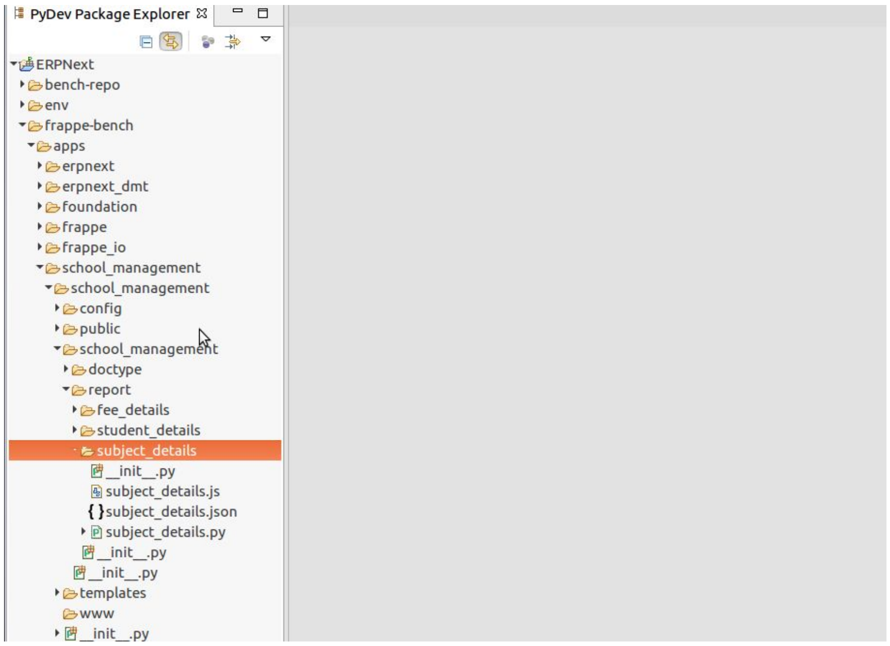
• Write a Python file in the same folder where this is saved and return column and result.

Ref DocType: Subject Management
Report Type: Script Report
Is Standard: Yes
Disabled: ☐
Apply User Permissions: ✓
Module: School Management
Add Total Row: ☐

	Role
1	System Manager
2	All

You edited this a minute ago
You created this

Script Report automatically created under the custom app:

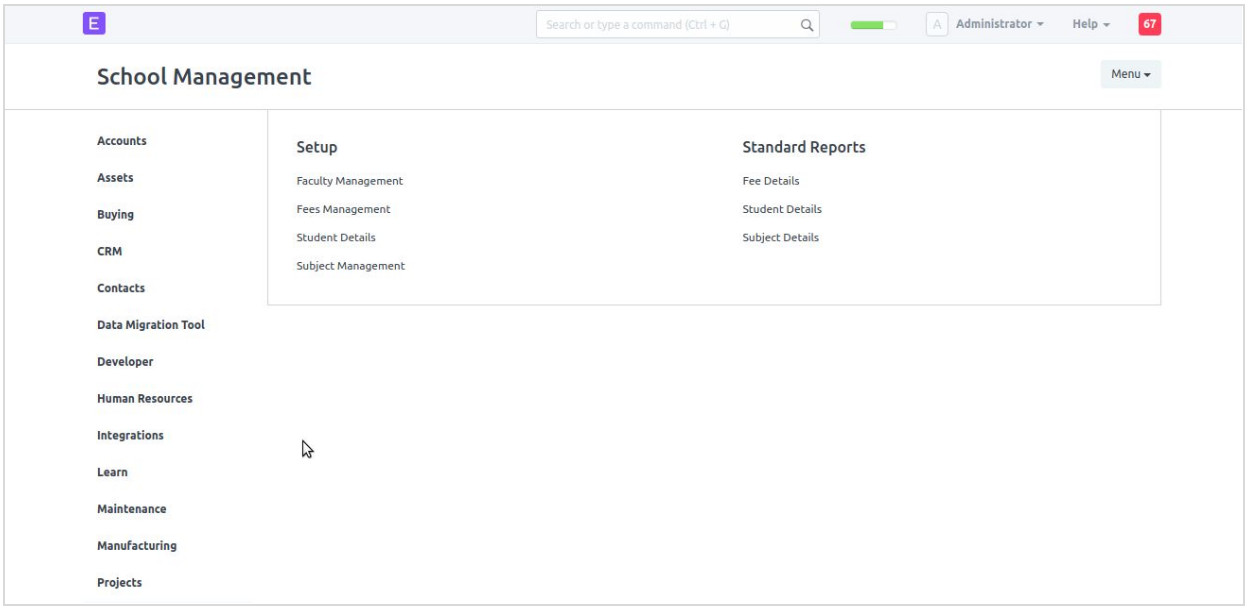

PyDev Package Explorer

```
ERPNext
  bench-repo
  env
  frappe-bench
    apps
      erpnext
      erpnext_dmt
      foundation
      frappe
      frappe_io
      school_management
        school_management
          config
          public
          school_management
            doctype
            report
              fee_details
              student_details
              subject_details
                __init__.py
                subject_details.js
                subject_details.json
                subject_details.py
              __init__.py
            __init__.py
          templates
          www
          __init__.py
```

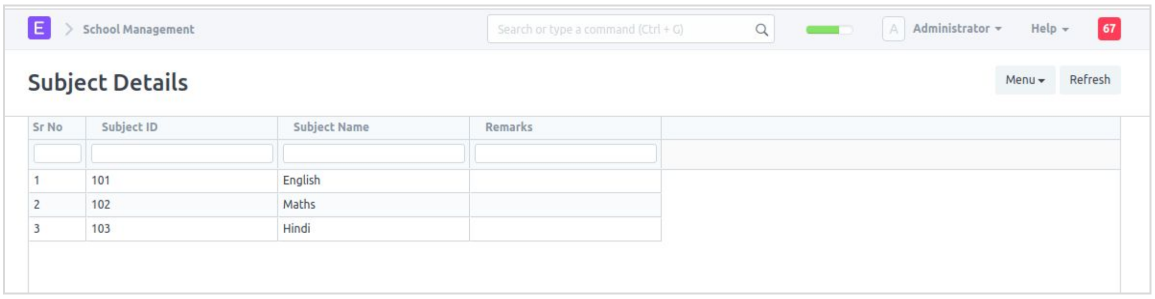
Write a script code in subject_details.py file.

```
# Copyright (c) 2013, solufy.in and contributors
# For license information, please see license.txt

from __future__ import unicode_literals
import frappe
from frappe import _

def execute(filters=None):
        columns = get_report_columns()
        data = get_report_data(filters)
        return columns, data

def get_report_columns():
        columns = [{
                "fieldname": "subject_id",
                "label": _("Subject ID"),
                "fieldtype": "Data",
                "width": 200
                },
                {
                "fieldname": "subject_name",
                "label": _("Subject Name"),
                "fieldtype": "Data",
                "width": 200
                },
                {
                "fieldname": "remarks",
                "label": _("Remarks"),
                "fieldtype": "Data",
                "width": 200
                },
        ]
        return columns

def get_report_data(filters=None):
        data = get_orders(filters)
        return data

def get_orders(filters):
        test_q = """select subject_id,subject_name,remarks
                from `tabSubject Management`"""
        return frappe.db.sql(test_q, as_dict=True)
```

User can access the subject Details Report under the Standard Reports section.



Report View



Sr No	Subject ID	Subject Name	Remarks
1	101	English	
2	102	Maths	
3	103	Hindi	

17. Helpful Command

General Usage:

bench –version - Show bench version

bench src- Show bench repo directory

bench --help- Show all commands and help

bench [command] --help- Show help for command

bench init [bench-name]- Create a new bench (Run from home dir)

bench --site [site-name] COMMAND- Specify site for command

bench update- Pulls changes for bench-repo and all apps, applies patches, builds JS and CSS, and then migrates.

- --pull Pull changes in all the apps in bench
- --patch Run migrations for all sites in the bench
- --build Build JS and CSS artifacts for the bench
- --bench Update bench
- --requirements Update requirements
- --restart-supervisor restart supervisor processes after update
- --upgrade Does major upgrade
- --no-backup Don't take a backup before update

bench restart Restart all bench services

bench backup Backup

bench backup-all-sites Backup all sites

- --with-files Backup site with files

bench restore Restore

- --with-private-files Restore site with private files (Path to tar file)

--with-public-files Restore site with public files (Path to tar file)

bench migrate Will read JSON files and make changes to the database accordingly

Config:

bench config - Change bench configuration

auto_update [on/off] Enable/Disable auto update for bench

dns_multitenant [on/off] Enable/Disable DNS Multitenancy

http_timeout Set http timeout

restart_supervisor_on_update Enable/Disable auto restart of supervisor

serve_default_site Configure nginx to serve the default site on

update_bench_on_update Enable/Disable bench updates on running bench...

bench setup - Setup components

auto-update Add cronjob for bench auto update

backups Add cronjob for bench backups

config overwrite or make config.json

env Setup virtualenv for bench

nginx generate config for nginx

procfile Setup Procfile for bench start

production setup bench for production

redis generate config for redis cache

socketio Setup node deps for socketio server

sudoers Add commands to sudoers list for execution...

supervisor generate config for supervisor add-domain add custom domain for site

firewall setup firewall and block all ports except 22, 80 and 443

ssh-port change the default ssh connection port

Development:

bench new-app [app-name] Creates a new app

bench get-app [repo-link] - Downloads an app from a git repository and installs it

bench install-app [app-name] Installs existing app

bench remove-from-installed-apps [app-name] Remove app from the list of apps

bench uninstall-app [app-name] Delete app and everything linked to the app (Bench needs to be running)

bench remove-app [app-name] Remove app from the bench entirely

bench --site [sitename] --force reinstall Reinstall with fresh database (Caution: Will wipe out old database)

bench new-site [sitename] - Creates a new site

--db-name Database name

--mariadb-root-username Root username for MariaDB

--mariadb-root-password Root password for MariaDB

--admin-password Administrator password for new site

--verbose erbose

--force Force restore if site/database already exists

--source_sql Initiate database with a SQL file

--install-app Install app after installation`

bench use [site] Sets a default site

bench drop-site Removes site from disk and database completely

--root-login

--root-password

bench set-config [key] [value] Adds a key-value pair to site's config file

bench console Opens a IPython console in the bench venv

bench execute Execute a method inside any app.

Eg :bench execute frappe.utils.scheduler.enqueue_scheduler_events

bench mysql Opens SQL Console

bench run-tests Run tests

--app App Name

--doctype DocType to run tests for

--test Specific Test

--module Run a particular module that has tests

--profile Runs a Python profiler on the test

bench disable-production Disables production environment

Scheduler:

bench enable-scheduler - Enables Scheduler that will run scheduled tasks

bench doctor - Get diagnostic info about background workers

bench show-pending-jobs- Get pending jobs

bench purge-jobs - Destroy all pending jobs

18. Field Types

Following are the types of fields you can define while creating new ones, or while amend standard ones.

Simple Fields:

Attach

Attach field allows you browsing file from File Manager and attach in the transaction.

Button

It will be a Button, on clicking which you can execute some functions like Save, Submit etc.

Check

It will be a checkbox field.

Column Break

Since ERPNext has multiple column layouts, using Column Breaks, you can divide the set of fields side-by-side.

Currency

Currency field holds a numeric value, like item price, amount etc. Currency field can have value up to six decimal places. Also, you can have a currency symbol being shown for the currency field.

Data

Data field will be a simple text field. It allows entering value up to 255 characters.

Date and Time

This field will give you date and time picker. Current date and time (as provided by your computer) is set by default.

Float

Float field carries numeric value, up to six decimal place. Precision for the float field is set in

Setup > Settings > System

Setting will be applicable to all the float field.

Image

Image field will render an image file selected in another attach field.

For the Image field, under Option (in Doctype), field name should be provided where the image file is attached. By referring to the value in that field, the image will be a reference in the Image field.

Int (Integer)

Integer field holds a numeric value, without a decimal place.

Geolocation

Use Geolocation field to store GeoJSON feature collection. Stores polygons, lines, and points. Internally it uses following custom properties for identifying a circle.

```
{
    "point_type": "circle",
        "radius": 10.00
    }
```
Password

Password field will have decode value in it.

Read Only

Read-only field will carry data fetched from another form, but they themselves will be non-editable. You should set Read Only as field type if its source for value is predetermined.

Section Break

Section Break is used to divide the form into multiple sections.

Select

Select will be a drop-down field. You can add multiple results in the Options field, separated by row.

Small Text

Small Text field carries text content, has more character limit than the Data field.

Text Editor

Text Editor is a text field. It has text-formatting options. In ERPNext, this field is generally used for defining Terms and Conditions.

Relational fields:

Link

Link field is connected to another master from where it fetches data. For example, in the Quotation master, Customer is a Link field.

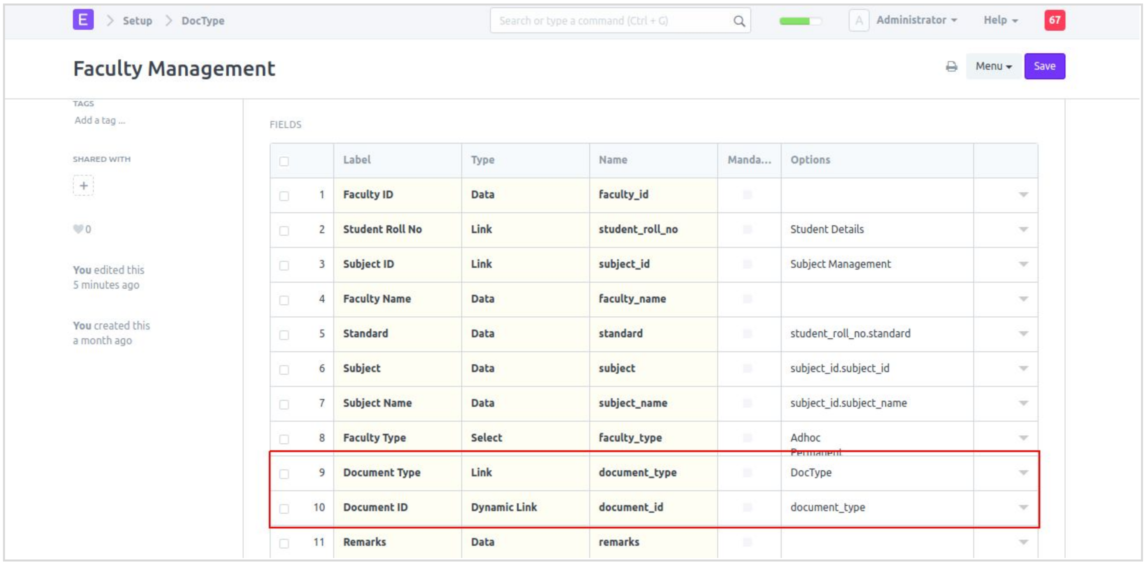
Table

Table will be (sort of) Link field which renders another doctype within the current form. For example, Item table in the Sales Order is a Table field, which is linked to Sales Order Item doctype.

Dynamic Link field is one which can search and hold the value of any document/doctype. Let's consider an example to learn how Dynamic Link field works.

Step 1: Insert Link Field for Doctype
Firstly we will create a link field which will be linked to the Doctype.

Step 2: Insert Dynamic Link Field
Insert “Document ID” dynamic link field. In this dynamic field set options above Doctype link field name.



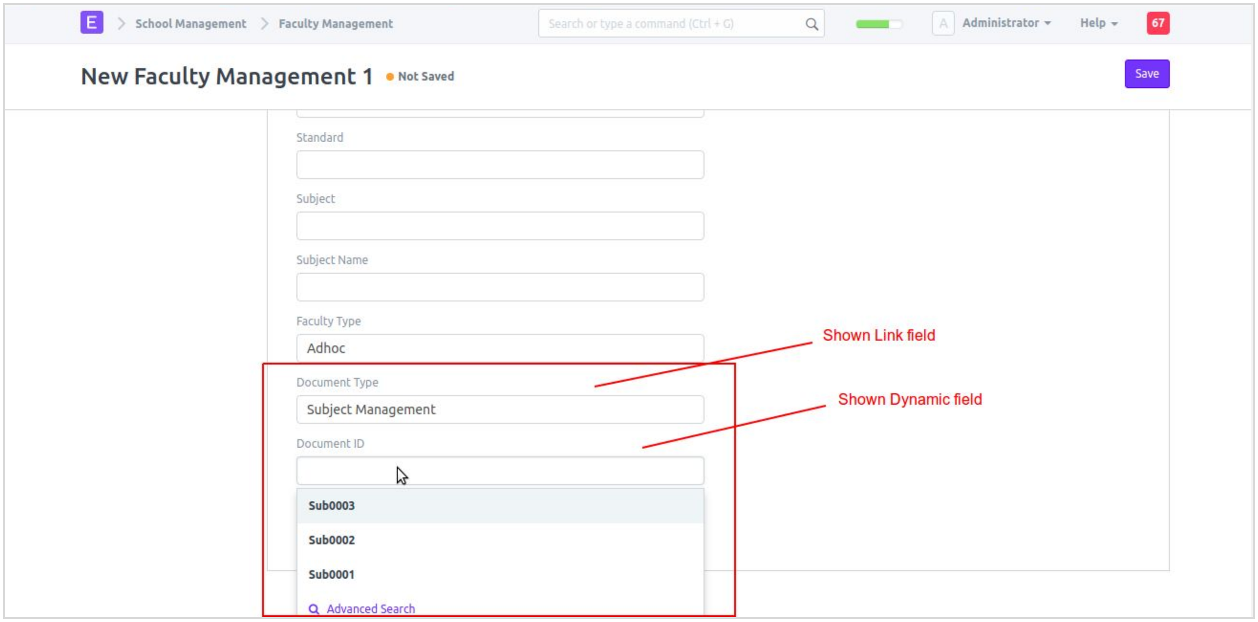
E > Setup > DocType

Faculty Management

FIELDS

	Label	Type	Name	Manda...	Options
1	Faculty ID	Data	faculty_id		
2	Student Roll No	Link	student_roll_no		Student Details
3	Subject ID	Link	subject_id		Subject Management
4	Faculty Name	Data	faculty_name		
5	Standard	Data	standard		student_roll_no.standard
6	Subject	Data	subject		subject_id.subject_id
7	Subject Name	Data	subject_name		subject_id.subject_name
8	Faculty Type	Select	faculty_type		Adhoc Permanent
9	Document Type	Link	document_type		DocType
10	Document ID	Dynamic Link	document_id		document_type
11	Remarks	Data	remarks		

Here, We can see doctype and Its related all the document ids so we can select any document id related to selected doctype. See the below snapshot.



E > School Management > Faculty Management

New Faculty Management 1 · Not Saved

Standard:

Subject:

Subject Name:

Faculty Type: Adhoc

Document Type: Subject Management

Document ID:

Sub0003

Sub0002

Sub0001

Advanced Search

19. Task Runner (Scheduled tasks)

Finally, an application also has to send email notifications and do other kind of scheduled tasks. In Frappe, if you have setup the bench, the task / scheduler is setup via RQ using Redis Queue.

To add a new task handler, go to hooks.py and add a new handler. Default handlers are all, daily, weekly, monthly, cron. The all handler is called every 4 minutes by default.

```
# Scheduled Tasks
# ---------------
scheduler_events = {
        "daily": [
                "school_management.tasks.daily"
        ],
        "cron": {
                "0/10 * * * *": [
                "school_management.task.run_every_ten_mins"
                ],
                "15 18 * * *": [
                "school_management.task.every_day_at_18_15"
                ]
        }
}
```

20. Form Client Scripting code using JS

Frappe.call
Callback

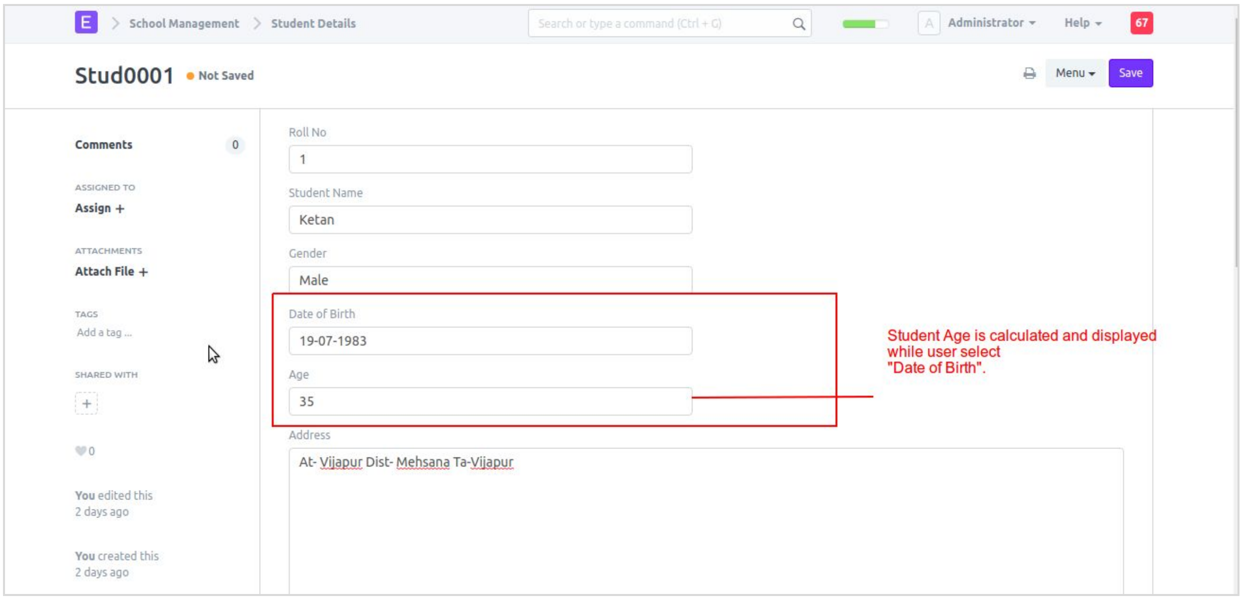
Here shown student age calculation while user select "Date of Birth" field at that time age calculation automatically calculated displayed in Age field.
Write a client side scripting code for student_details.js file.

```
// Copyright (c) 2018, solufy.in and contributors
// For license information, please see license.txt

frappe.ui.form.on('Student Details', {

        date_of_birth: function(frm) {
                return frm.call({
                        method:
                        "school_management.school_management.doctype.student_details.
                         student_details.get_age",
                        args: {
                                date_of_birth: frm.doc.date_of_birth
                        },
                        callback: function(r)
                        {
                                console.log("This is callback response",r.message);
                                frm.set_value("age", r.message);

                        }
                });
        },
});
```

Write a server side scripting code for student_details.py file.

```
# -*- coding: utf-8 -*-
# Copyright (c) 2018, solufy.in and contributors
# For license information, please see license.txt

from __future__ import unicode_literals
import frappe
from frappe.model.document import Document
import datetime
from dateutil.relativedelta import relativedelta


class StudentDetails(Document):
        pass

@frappe.whitelist()
def get_age(date_of_birth=None):
        print "call python method::::::::"
        age = 0.0
        if date_of_birth:
                d1 = datetime.datetime.strptime(date_of_birth, "%Y-%m-%d").date()
                print "d1 ::::::::::::::",d1
                d2 = datetime.datetime.today().date()
                print "d2 ::::::::::::::",d2
                rd = relativedelta(d2, d1)

                print "Display Age",rd
                rd.years

                age = rd.years or 0.0
        return age
```

Student age calculation and displayed age field while user select “Date of Birth” field. See the below snapshot:



Roll No: 1

Student Name: Ketan

Gender: Male

Date of Birth: 19-07-1983

Age: 35

Address: At- Vijapur Dist- Mehsana Ta-Vijapur

Student Age is calculated and displayed while user select "Date of Birth".

21. References

https://frappe.io/docs/user/en/tutorial