# School Portal API Specifications

## Users

1. Student
2. Teacher
3. Parent
4. Principal
5. Admin

## Entities in the system

1. User
   1. We extend Django User class adding sex, and type (this determines whether they are Admin, Student, etc.)
   2. Based on the user type we create Student, Teacher, Parent, Principal, etc.
2. Student
   1. One to one field with user
   2. Date of birth
   3. Address
   4. Parent (foreign key)
   5. Class (foreign key)
   6. School (foreign key)
3. Teacher
   1. One to one field with user
   2. Date of birth
   3. Address
   4. Subject (foreign key)
4. Class Teacher
   1. One to one field with teacher
   2. Class (foreign key)
5. Parent
   1. One to one field with user
   2. Address
   3. Student (foreign key)
6. Principal
   1. One to one field with user
   2. School (foreign key)
7. Admin (nothing the user type is enough)
8. Class
   1. Name
   2. School (foreign key)
9. School
   1. Name
   2. Address
10. Subject
    1. Name
    2. Type (compulsory or optional)
    3. CA (max score u can have in d subject CA)
    4. Exam (max score u can have in d subject Exam)
11. Term
    1. Term, (Integer, 1 = 1st term, etc.)
12. Session
    1. Session (charfield e.g. 2016/2017)
13. Result
    1. Student (foreign key)
    2. Subject (foreign key)
    3. Class (foreign key)
    4. Term (integer)
    5. Session (foreign key)
    6. CA
    7. Exam
    8. Final
    9. Comment (text)
14. Position (per term based on where the student ranks when his total score in all subjects are compared to that of students in his/her class)
    1. Student (foreign key)
    2. Class (foreign key)
    3. Term (integer)
    4. Session (foreign key)
    5. Position
    6. Comment (text, by class teacher)

## Server Functions

* We are using Django Rest Framework
* Login (Token Auth via JWT)
  + We are using Django Rest Framework JWT <http://getblimp.github.io/django-rest-framework-jwt>
  + The instructions on d website for implementing it are straightforward
* Admin Access to all entities (duh… its django)
* Chat rooms based on teacher + students taking a subject in a class
  + Also based on class teacher + students in the class

## API Endpoints

* Student, Parent, Teacher, Principal, Subject, Class, School, Term, Session (Read, Update)
* Class Teacher, Result, Position (Create, Read, Update, Delete)

## Client Functions

* Teacher access to upload scores of students for their subject and comment on student performance
* Student/Parent access to their result and position info
* Principal access to
  + Assigned to school
  + Info of teachers
  + Info of students
  + Result info of all students in school
* Class Teachers computes results for students in his/her class and comments on students performance.