**School Placement System Algorithm**

1. Start

**//USER FEATURE (SOFTWARE ADMINSTRATORS)**

2. Input whether **USER** **type\_of\_user** is new or existing

3. IF User is new

A) ASK if user is creating a high School admin or Student profile

B) INPUT user’s option

C) IF it is a high School admin profile

Input School full name, abbreviation name, location of region

GENERATE high School number and password with details provided

INSTANTIATE object of high School admin profile

SAVE object data Model for high School admin profile

DISPLAY profile of high School admin

ASK if **USER** wants to login or exit

IF **USER** wants to login

Go to step 4

IF User wants to exit

Break program

D)  IF it is a Student profile

INPUT surname, other names, index number

INSTANTIATE object of **Student** profile and **Results**, **Application** with null values (Results and Students profile should be fixed while there is flexibility for the Application Model to be modified)

SAVE object data Models for Student profile

DISPLAY profile of Student

ASK if USER wants to login or exit

If User wants to login

Go to **step 4**

If User wants to exit

Break program

**(END USERS)**

4. IF **USER** is existing

A) ASK if user is a high School admin or Student profile.

B) CASE user is a high School admin

ASK user for school number and password

IF there is a School object with the credentials provided

INSTANTIATE the School profile Object model with the stored values of the attributes

DISPLAY “Welcome to the School Admin Dashboard”

**//School Admin Features**

ASK the user which feature to open

Input feature option from User

IF feature option is “Courses”

//**Courses** feature

DISPLAY the Course operation options

INPUT course operation option

CASE where course operation option is “**Create** New Course”

INPUT course name, cut off raw score, max number of enrollment

INSTANTIATE an object of Course with instance attributes course name, school code, base raw score, max enrollments.

GENERATE the value of course code attribute

Save Object Model

IF Course object Model is saved

GO TO Course operation options menu

DISPLAY “New Course has created successfully”.

CASE where course operation option is “**View** Courses”

GET the school code of the admin.

RETRIEVE all saved Course objects with that school code.

DISPLAY the retrieved Course object details including course names, course code, base raw score,max enrollments.

DISPLAY “Input a key to go back”

IF key is pressed

GO TO Course operation options menu

CASE where course operation option is “**Delete** Course”  
 DISPLAY ”Kindly enter in the the course code of the Course you want to delete”

INPUT Course code from USER

RETRIEVE saved Course object Model

DELETE object model with the course code attribute value entered by USER.

GO TO Course operation options menu

DISPLAY “Course is successfully deleted”

CASE where course operation option is “**Update** Course Details”

DISPLAY ”Kindly enter in the the course code of the Course you want to update”

INPUT Course code from USER

RETRIEVE saved Course object Model that matches the course code

ASK USER which aspect to update(course name, base raw score, max enrollment and go back)  
 INPUT User choice

IF User choice is “course name”

INPUT new course name

REASSIGN course name attribute value of the Course to new course name.

ELSE IF User choice is “base raw score”

INPUT new base raw score

REASSIGN course name attribute value of the Course to new base raw score

ELSE IF User choice is “max enrollment”

INPUT new course name

REASSIGN course name attribute value of the Course to new max enrollment.

CASE where course operation option is “Go Back”

GO TO School admin main menu

IF feature option is “School Profile”

**//School Profile feature**

CASE where School Profile operation option is “**View** School Profile”

GET the school code of the admin.

RETRIEVE all the data of the School profile.

DISPLAY the retrieved School object details including School full name, abbreviation name, location of region, school number, password.

DISPLAY “Input a key to go back”

IF key is pressed

GO TO School Profile operation options menu

CASE where course operation option is “**Update** School Profile”

` GET the school code of the admin.

RETRIEVE all the data of the School profile.

INSTANTIATE an object model of School profile

ASK USER which aspect to update(School full name, abbreviation name, location of region, password)  
 INPUT User choice

IF User choice is “school full name”

INPUT new School full name

REASSIGN School full name attribute value of the School profile to new School full name.

ELSE IF User choice is “abbreviation name”

INPUT new abbreviation name

REASSIGN abbreviation name attribute value of the School to new base raw score

ELSE IF User choice is “location of region”

INPUT new course name

REASSIGN course name attribute value of the Course to new max enrollment.

IF feature option is “Applications”

-Retrieve

-Update Admission status.

-

Nb:give success, error and warning alerts

     Catch the expections in different processes

      Use clear screen, return and looping of menus for better user experience

**STUDENT SECTION.**

C) CASE USER is a Student

ASK **USER** for Student index number and password

IF there is a **Student** object with the credentials provided

ASSIGN the current Student profile Object model to a variable

DISPLAY “Welcome to the Student Dashboard”

ASK the user which feature to open

Input feature option from User

IF feature option is “My Results”

//**Result** features

ASK what operation to do on Results

IF operation option is “**View** Results”

RETRIEVE Result object with attributes (course name, code, score, grade)of current User

DISPLAY details of Result object with details Course name, code, score, grade

INPUT an key from User

IF key is entered

GO TO Results feature menu

//**Application** feature

IF feature option is “Application”

ASK what operation to do on Application

IF operation option is “Create Application" IF operation option is “**View** Application”.

RETRIEVE Application object with attributes (index number, code, score, grade)of current User

DISPLAY details of Application object with attributes Course name, code, score, grade

INPUT an key from User

IF key is entered

GO TO Results feature menu

ELSE IF operation option is “**Update** Choices”

RETRIEVE Result object with attributes (course name, code, score, grade)of current User

ASK User to input Subject code

INPUT subject code from User

ASK Student USER to enter new score for subject

INPUT new score for the subject

COMPUTE grade for the score

UPDATE score and grade with new values

CALL view results method

//Admission Status features

View

**//School selection Info**

Retrieve

**UNIFIED MODELLING LANGUAGE**

|  |
| --- |
| Application |
| - index\_number:int  -int first\_choice[3] -school,course,admission status  - int second\_choice[3]  - int third\_choice[3]  - int fourth\_choice[3] |
| +Application() constructor  +view\_application(int index, Application \*application) : void  +view\_applications(index\_number,school number) : void  +update\_application(index\_number  ,choice) : void  +school\_placement(school\_id) : void  +set\_application(int [],int [], int [], int [] ) : void  +create\_application(int index\_no)  +update\_admission\_status(int school\_id) : void |

|  |
| --- |
| Student |
| -Surname: string  -Othernames :string  -Index number:int  -Password: string  -rawscore |
| +Student() :constructor   +display\_student(Student\*)  +verify\_student(int,.string,Student\*) |

-

|  |
| --- |
| Course |
| -course\_id:int  - course\_name :string  - school\_id: int  - max\_enrollment: int  - base score:unsigned int |
| +Course(all attributes):constructor  +view\_course(school\_id):  +view\_course(course\_id, school\_id):  -delete\_course(course\_id)  -update\_course(course\_id,option,  new\_value) |

|  |
| --- |
| Result |
| - Index\_number:int  -subject\_1:int array{score,grade}  -subject\_2:int array{score,grade}  -subject\_2:int array {score,grade}  -subject\_3:int {score,grade}  -subject\_4:int {score,grade}  -subject\_5:int {score,grade}  -subject\_6:int {score,grade}  - raw\_score:int |
| +Result()  +set\_result(index\_numbber,subject\_1,….,subject\_6)  +Display\_result(index number)const:  - compute\_totalscore(): |

|  |
| --- |
| School Profile |
| school\_id : int  school\_name : string  school\_prefix : string  password: string |
| +school(all attributes)  +display\_school()  +display\_school(school\_id) |