School System – test results

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S/No | General | | | | | | | | | | |
|  | Consistency | | | | | | | For all forms, show consistency. E.g.   1. Let control buttons i.e crud buttons be at the right bottom are of the form all the time 2. On hitting enter/cancel, all forms should behave the same 3. On hitting cancel, all forms should ask for saving unsaved data 4. All forms that list a table, should show the primary key. 5. All input fields should be validated before trying an add or update 6. All ‘adds’ must first confirm the record does not exist. 7. All deletes must be confirmed first 8. All foreign keys must be lookups(preferably through combobox) unless the expected number is too large in which case you should use select( :: )button 9. All errors in a **form** must be trapped and show to the user. Show the error message as well as the innerexception message if exists 10. All errors in a **library** must be trapped and logged. 11. All data entry forms should be validated according to the database structure. E.g. where data is nullable, don’t make the textbox mandatory | | | |
|  | Upload/download | | | | | | | Remove in all forms. Create an upload/download wizard that   1. Uploads to a specific table 2. Should take care of constraints –eg primary key, unique indexes, foreign key, mandatory field 3. Should take care of data types | | | |
|  | Login screen | | | | | | | | | | |
|  | Use integrated security | | | | | | | | If unchecked, show a place to key in database login username and password and use them to connect | | |
|  |  | | | | | | | |  | | |
|  | Main Form | | | | | | | | | | |
|  | Status bar | | | | | | | | If you double click selected database, show a form that shows the system details | | |
|  |  | | | | | | | |  | | |
|  | Programmes | | | | | | | | | | |
|  | Edit🡪programme courses 🡪 delete | | | | | | Not deleting | | | | |
|  | Programme Years | | | | | | Introduce a programmme year table with cols –ProgrammeId, Year(int), Description(text), NextYr(int). The primary key is programmeId + Year. | | | | |
|  | Change the table Programme course to Programme Year Courses | | | | | | Introduce a programmme year courses table with cols –ProgrammeId, Year(int) CourseId, Semester(int). The primary key is programmeCourseId. | | | | |
|  |  | | | | | |  | | | | |
|  | Teacher | | | | | | | | | | |
|  | Add | | | | | | Remove mandatory on ‘Position’; ‘date joined’ | | | | |
|  | Edit🡪Teacher | | | | | | Date left cannot be less than date joined | | | | |
|  |  | | | | | |  | | | | |
|  | Classes | | | | | | | | | | |
|  | Add | | | | | | Remove mandatory on ‘Position’; ‘date joined’ | | | | |
|  | Edit🡪Teacher | | | | | | Date left cannot be less than date joined | | | | |
|  |  | | | | | |  | | | | |
|  | Database control panel | | | | | | | | | | |
|  | Connect to server | | | | | | Make login name and pwd optional where integrated security is selected | | | | |
|  | Code | | | | | | 1. Replace all the code in this form with the equivalent in the sqlhelper class that I created 2. Remove all references to db.SBServers. infact delete sbserver table | | | | |
|  | Update xml | | | | | | Create a List<SBSystem> systems from the databases listed then  Use SQLHelper.SaveXML(List<SBSystem> systems, string filename) | | | | |
|  | Set as default | | | | | | Use SQLHelper.SaveXML(List<SBSystem> systems, string filename) where the default system is now selected | | | | |
|  | Restore | | | | | | When restore is clicked, bring up a form that asks restore to a) same place or b) different place. Same places uses RestoreDataBase(string BackupFilePath,  string destinationDatabaseName,  string DatabaseFolder,  string DatabaseFileName,  string DatabaseLogFileName)  and different place  BackupDatabase(SqlConnectionStringBuilder csb, string destinationPath) | | | | |
|  | code | | | | | |  | | | | |
|  | School | | | | | | | | | | |
|  | Edit | | | | | | Only one school should be default. If one school default = true, all other schools default should be set to false. | | | | |
|  | Default | | | | | | Not working. Cant click | | | | |
|  | School🡪Delete | | | | | | Prevent deleting if there are students associated with it | | | | |
|  | Classes | | | | | | | | | | |
|  | Programme | | | | | | Connect the class to ProgrammeYear and not programme | | | | |
|  | Shortcode | | | | | | To be unique id | | | | |
|  | Add | | | | | | The programme combo to contain ProgrammeYear and not programme | | | | |
|  | Class🡪subjects | | | | | | Change it into a datagrid. To populate from programmeYearCourses for the user to enter course teachers & status only. | | | | |
|  | Class🡪subjects🡪Delete | | | | | | Error on delete. The message is not correct. It is bringing the message even when the class is new | | | | |
|  | Students | | | | | | | | | | |
|  | Students List | | | | | | A listbox cannot hold a lot of records, it either crashes or will take too long to load and will freeze for the time it is loading. Since students are expected to be in hundreds or thousands, you need to re-design this as an intelligent select using selected key fields such as admin no, class, surname, etc | | | | |
|  | School | | | | | | Should pre-select the default school only | | | | |
|  | Edit🡪Account | | | | | | Create account not working | | | | |
|  |  | | | | | |  | | | | |
|  | Grading system | | | | | | | | | | |
|  | Add | | | | | | Show the added item immediately | | | | |
|  | Delete | | | | | | Refresh the listbox immediately | | | | |
|  | Student - edit | | | | | | | | | | |
|  |  | | | | | | Add date of admission | | | | |
|  | Extra curriculum | | | | | | Add a tab for editing students extra curriculum | | | | |
|  | Parents info tab | | | | | | Group the details into father info; mother info; guardian info | | | | |
|  | Residence | | | | | | Introduce a table for residence – i.e. physical address of where the student is residing; if in campus then include halls of residence | | | | |
|  | Special needs | | | | | | Introduce a tab. It may have info on donors, ailments, food, allergies, etc | | | | |
|  | Disciplinary | | | | | | Introduce a tab to keep disciplinary issues; date; incident; incident category; severity; action recommended; action taken; discipline rating; review | | | | |
|  | Time Table | | | | | | | | | | |
|  | On Double click | | | | | | If the date/time has no item bring up the form “TimeTableDetailsForm” and create a new entry  If the date/time has an item bring up the form “TimeTableDetailsForm” and edit the entry | | | | |
|  | TimeTableDetailsForm | | | | | | On recurrent combo 🡪 add weekly | | | | |
|  | Class stream | | | | | | When the selection changes, the time table should change | | | | |
|  | Save | | | | | | To save to database | | | | |
|  | Print | | | | | | Include a button for printing the selected timetable | | | | |
|  | Printall | | | | | | Include a button for printing all time-tables | | | | |
|  | Onload | | | | | | Should show the items that are recurring if within the startdate and enddate | | | | |
|  | Scroll | | | | | | So as to see the timetable between 8 and 5 am | | | | |
|  | TimeTableDetailsForm | | | | | | 1. Subjects 🡪 to only see the subjects of a given stream. However one can add any other activity 2. Rename subject to activities 3. Rename room to venue, one can add any other venue not necessarily a room. 4. Add 🡪 check conflicts | | | | |
|  | Export/import | | | | | | Export to excel, word, xml | | | | |
|  | Exam Types | | | | | | | | | | |
|  | Edit | | | | Add % contribution field | | | | | | |
|  | Exam period 🡪 Table has been split into **Exam** and **ExamExamTypes** | | | | | | | | | | |
|  | Form Title | | | | Rename to exam-type period | | | | | | |
|  |  | | | |  | | | | | | |
|  | Subjects | | | | | | | | | | |
|  | Hrs | | | | Add a field for number of required hours | | | | | | |
|  | Fees | | | | Add a field for subject amount | | | | | | |
|  | Classes | | | | | | | | | | |
|  | Class stream | | | | All classes must have at least one(1) stream | | | | | | |
|  | Programme | | | | | | | | | | |
|  | Hrs | | | | Add a field for number of required hours | | | | | | |
|  | Fees | | | | Add a field for subject amount | | | | | | |
|  | Programme Years | | | | | | | | | | |
|  | Hrs | | | | Add a field for number of required hours | | | | | | |
|  | Fees | | | | Add a field for subject amount | | | | | | |
|  | Programme Year Courses | | | | | | | | | | |
|  | Hrs | | | | Add a field for number of required hours | | | | | | |
|  | Fees | | | | Add a field for subject amount | | | | | | |
|  | Select exam form | | | | | | | | | | |
|  | Datagrid | | | | Black. Correct | | | | | | |
|  | Operator combo | | | | When a field is selected, show only those operators that can work with that datatype. E.g. int does not work with sartswith op | | | | | | |
|  | Value field | | | | Create the value field according to type. Eg. Create a combox if the type is a look up etc.  Example if field is gender, then value field should be a combobox with M/F | | | | | | |
|  |  | | | |  | | | | | | |
|  | Register Exams | | | | | | | | | | |
|  | Form | | | | Add a contribution % field | | | | | | |
|  | Save | | | | 1. On validation, make sure that total contribution for all exams in that Year/Term add up to 100% 2. Make sure the contents of grid are saved – e.g. when a record is deleted, save should delete it from database 3. Trap errors such as ‘delete foreign key first’ | | | | | | |
|  | View contribution | | | | Add a button to view contributions for that exam time. | | | | | | |
|  | Create Exam Timetable | | | | 1. Create a ‘Create Exam Timetable’ menu to create an exam timetable for the selected class 2. Create ‘Create All Exam Timetable’ menu to create an exam timetable for all classes | | | | | | |
|  | Populate | | | | Messgabox(Done) | | | | | | |
|  | Status | | | | Put a combo to edit status | | | | | | |
|  | Exam Rules | | | | | | | | | | |
|  | Form | | | | Add a table for exam rules. The table will help create a) exam registration ruleset object and b) exam processing ruleset object. The ruleset objects shall be used by the student registration form to allow or disallow registration. | | | | | | |
|  | CRUD | | | | Crud forms for editing ruleset | | | | | | |
|  | Register Students | | | | | | | | | | |
|  | Populate | | | | | Should check the examination registration ruleset and populate only those that have passes | | | | | |
|  | Examination Cards | | | | | Allow for printing of examination card labels for duly registered students | | | | | |
|  | Populate | | | | | Refresh the grid after populate | | | | | |
|  | Grid | | | | | Show status | | | | | |
|  | Subject combobox | | | | | Should show subjects for registered exams that status = open and enabled **only**. I.e you cannot register for a closed registered exam | | | | | |
|  | Access | | | | | This form should be accessible to only those who are allowed. This should include selection of period, class, subject | | | | | |
|  |  | | | | |  | | | | | |
|  | Marksheet | | | | | | | | | | |
|  | Subject combobox | | | | | | Should show subjects for registered exams that status = open and enabled **only**. I.e you cannot register for a closed registered exam | | | | |
|  | Access | | | | | | This form should be accessible to only those who are allowed. This should include selection of period, class, subject | | | | |
|  | Save | | | | | | Saves changes | | | | |
|  | Commit | | | | | | Introduce a commit button. When committed, the registered exam should not be accessible again  Update lastmodified, modified by fields | | | | |
|  | Grid | | | | | | Allow sorting and filtering by the clicking or right-clicking on the column header | | | | |
|  | Process exams | | | | | | | | | | |
|  | Selection | | | | | | By a) ExamId/Class /Subject b) ExamId /Class c) ExamId | | | | |
|  | Processing Activities | | | | | | A) ExamId/Class /Subject  using the selected subject do   1. Selection ExamId🡪Class🡪Subject 2. Make sure Exam.status == open 3. Make sure ExamExamTypes.Contribution == 100% 4. For each registered student    1. Combine their marks for cat, main exam etc as follows FinalMark = sum(ExamContributionPercent \* Mark)    2. If FinalMark>100 throw exception    3. Look up the grade of FinalMark    4. Clear any record that may be there in studentexamresultdetail    5. If not existent add an incomplete record in studentexamresult    6. Enter new record in the result in studentexamresultdetail table    7. Update/complete the record in studentexamresult   B) ExamId /Class  using the selected class do   1. Selection ExamId🡪Class 2. Make sure Exam.status == open 3. Studentexamresult.Delete(p=>p.ExamId == examid && p.Class == classid). Cascade delete to studentexamresultdetail 4. For each student in class    1. Add an incomplete record in studentexamresult    2. Save to get studentexamresult.Id    3. For each registered subject in that class       1. Do (A) above    4. Compute mean grades    5. Update/complete the record in studentexamresult   C) Year/Term/  using the selected class do   1. Clear studentexamresult and studentexamresultdetail for the selected ExamId 2. For each programme do    1. For each class in programme do       1. Do ‘B’ above | | | | |
|  | Access | | | | | | Only to authorized users | | | | |
|  | Reports – lets have reports by various subgroups  Reports - Generic list dumps | | | | | | | | | | |
|  | Name | | | | | | Purpose | | | | Details |
|  | Table lists (e.g. student lists, class lists, subject lists, etc) | | | | | | To output information already stored in the system | | | | All tables should be listed in one way or another. Have a generic wizard of outputting a list. The wizard would have(see access for an example)   1. Select table(s) to retrieve info from 2. Select fields to list 3. Put a **where** criteria 4. Put a **sort** criteria 5. Put a **group by** criteria 6. Output to screen, excel, etc |
|  | Reports - Academic Report | | | | | | | | | | |
|  | Student academic reports | | | | | | Report form | | | | Shows student performance for **one** exam |
|  |  | | | | | | Progress form | | | | Shows student performance for **given a range of dates or exams** exam |
|  |  | | | | | | Student performance by target | | | | Shows student performance of **one** exam against **own/set/agreed** performance target |
|  |  | | | | | |  | | | |  |
|  | Class academic reports | | | | | | Report form | | | | Shows student performance for **one** exam |
|  |  | | | | | | Progress form | | | | Shows student performance for **given a range of dates or exams** exam |
|  |  | | | | | | Student performance by target | | | | Shows student performance of **one** exam against **own/set/agreed** performance target |
|  |  | | | | | | Class consolidated marksheet | | | | Tabular marksheet for subjects / students |
|  |  | | | | | |  | | | |  |
|  | Teacher academic reports | | | | | | Report form | | | | Shows student performance for **one** exam as measured by student performance |
|  |  | | | | | | Progress form | | | | Shows student performance for **given a range of dates or exams** exam as measured by student progress |
|  |  | | | | | | Teacher performance by target | | | | Shows student performance of **one** exam against **own/set/agreed** as measured by students performance against target |
|  |  | | | | | |  | | | |  |
|  | School Academic Reports | | | | | | Report form | | | | Of the whole school |
|  |  | | | | | | Progress form | | | | Shows school performance for **given a range of dates or exams** exam |
|  |  | | | | | | School performance by target | | | | Shows school performance of **one** exam against **own/set/agreed** performance target |
|  |  | | | | | | School performance in the region | | | | School performance against others in the region |
|  |  | | | | | |  | | | |  |
|  | Reports - Non Academic Report | | | | | | | | | | |
|  | Student academic reports | | | Discipline Status | | | | | | Shows student’s discipline status | |
|  |  | | | Disciplinary Record | | | | | | Shows student’s statement of disciplinary incidents | |
|  |  | | | Extra curricula | | | | | |  | |
|  |  | | | Medical record | | | | | |  | |
|  |  | | | Attendance record | | | | | |  | |
|  | Reports - Financial Reports | | | | | | | | | | |
|  | Student reports | | | Student statement | | | | | | Shows transactions for a given period | |
|  |  | | | Student Account status | | | | | | Shows student’s account balances | |
|  |  | | | Arrears / Receivables | | | | | | Shows students in arrears by class | |
|  |  | | | Parent statement | | | | | | Shows students statements for all children | |
|  |  | | | Parent Account status | | | | | | Shows Parent’s account balances | |
|  |  | | | Payment receipts | | | | | |  | |
|  |  | | | Fees structure | | | | | |  | |
|  | School financial reports | | | GL | | | | | |  | |
|  |  | | | PL | | | | | |  | |
|  |  | | | BS | | | | | |  | |
|  |  | | | Bank Statement | | | | | |  | |
|  |  | | | CashBook | | | | | |  | |
|  |  | | | Arrears / Receivables | | | | | |  | |
|  |  | | | Payables | | | | | |  | |
|  | Fees structure | | | By class | | | | | |  | |
|  |  | | | By Programme | | | | | |  | |
|  |  | | | By Account Types | | | | | |  | |
|  | Settings | | | | | | | | | | |
|  | Data | Remove all settings that don’t relate to school | | | | | | | | | | | |
|  |  |  | | | | | | | | | | | |
|  | Database Control Panel | | | | | | | | | | | |
|  | Start up | | Taking too long to show, warn user or just show it without initialization | | | | | | | | | |
|  |  | |  | | | | | | | | | |
|  | Accounts | | | | | | | | | | | |
|  | Start up | Since accounts table can be large, start up with a selection screen, just like students. | | | | | | | | | | | |
|  | Add | Rename title to “Open account”  Not working  Remove fields book balance, cleared balance, accrued interest, limit, interest rate, pass flag  School branch – disable the control & default to the default school | | | | | | | | | | | |
|  | OwnerId | Listbox to show recently accessed customers, | | | | | | | | | | | |
|  | Fees structure | Introduce a form for posting fees. Most schools have a fee-structure form. | | | | | | | | | | | |
|  | Fees Payment | Introduce a form for posting fees payment. | | | | | | | | | | | |
|  | About | | | | | | | | | | | |
|  | License Info | Create a form for showing license info (licensed to, customerno, customer name, email, contact, application, serial number and expiry date). | | | | | | | | | | | |
|  | Request License | Include a button for requesting for a license. This creates the xml shown below. The xml is then sent to softwareproviders either through email or sms  <license>  <licenseaction>renew</licenseaction>  <CustomerNo>Mbugus</CustomerNo>  <CustomerName>Mega Tech</CustomerName>  <Email>[mega@yahoo.com</Email](mailto:mega@yahoo.com%3c/Email)>  <Contact>Julius Muraya</Contact>  <AppId>ef0c7bbe-9567-45cb-b827-3269f3e53250</AppId>  <Address>44720 NRB</Address>  <SerialNo>1</SerialNo>  <Expires>1/5/2013 12:00:00 AM</Expires>  </license> | | | | | | | | | | | |
|  | Read License | Softwareproviders returns a signed license file called license.lic that looks like below. The signed file is read by the sb system whenever it wakes up.  <license>  <licenseaction>renew</licenseaction>  <CustomerNo>A100</CustomerNo>  <CustomerName>Mega Tech</CustomerName>  <Email>[mega@yahoo.com</Email](mailto:mega@yahoo.com%3c/Email)>  <Contact>A . N Other</Contact>  <AppId>ef0c7bbe-9567-45cb-b827-3269f3e53250</AppId>  <Address>44720 NRB</Address>  <SerialNo>1</SerialNo>  <Expires>1/5/2013 12:00:00 AM</Expires>  <Signature xmlns="[http://www.w3.org/2000/09/xmldsig#](http://www.w3.org/2000/09/xmldsig)">  <SignedInfo>  <CanonicalizationMethod Algorithm="<http://www.w3.org/TR/2001/REC-xml-c14n-20010315>" />  <SignatureMethod Algorithm="<http://www.w3.org/2000/09/xmldsig#rsa-sha1>" />  <Reference URI="">  <Transforms>  <Transform Algorithm="<http://www.w3.org/2000/09/xmldsig#enveloped-signature>" />  </Transforms>  <DigestMethod Algorithm="<http://www.w3.org/2000/09/xmldsig#sha1>" />  <DigestValue>ziQzbLO1sqBInziH2e8Upmf+kTg=</DigestValue>  </Reference>  </SignedInfo>  <SignatureValue>oZYHiKVDm0wzYD2B0JKk1r+cCCoJYL5dj+rQUpwe6QBRYKFZkMIyHm19PXYKM3MDscPa1zR0XEGT4yhvYzHUqq+fAkrRi66zFrhqtPAyxX0BqI3LP9vDak+ybmS5ZN+w9LeAgVtZ+twYCz0sTpE5DPW74WJm9AlIKI4tsqv64Zk=</SignatureValue>  </Signature>  </license> | | | | | | | | | | | |
|  | Help | | | | | | | | | | | |
|  | Help File | Create a help HTML file that illustrates ‘how to…’ functions and features of the whole system. | | | | | | | | | | | |
|  | Users, roles and rights | | | | | | | | | | | |
|  | Roles | Create a form for roles and CRUD functions. Don’t hard code in combobox in the users form | | | | | | | | | | | |
|  | Objects | Create a form for roles and CRUD functions. Don’t hard code in combobox in the users form | | | | | | | | | | | |