An Open Source Solution for Classroom Management EduXes

V Master on Free Software Projects Development and Management 2011-2012

— Author —
— José Antonio Salgueiro Aquino —
<info@joseantonio.org>

—Tutor—
Manuel Rego Casasnovas

December 28, 2012

Copyright

Copyright ©2012 José Antonio Salgueiro Aquino. Some rights reserved. This work is free and is licensed under the conditions of Creative Commons Attribution - Share Alike 3.0 Unsupported license. You can use, distribute and reuse this work if the same license is applied and the author is quoted Full text of the license can be read on http://creativecommons.org/licenses/by-sa/3.0/deed.en

Acknowledgments

Acknowledgments to my mentor, Manuel Rego. To Ramón Castro Pérez who helped me to run Siestta work. To José Manuel Ciges Regueiro, which memory I used as a template for my own work, and every 5th Master Software Libre classmate.

Description of the practicum

The main objective of this Master Thesis consist in the development of an mobile application to be used int highschools by teachers. It could allows teachers to carry on control students attendance, their behavior. Also it permits quick assessment by activity. Teachers would read students reports: weekly and daily assessment, by activity assessments and total marks.

- Name : José Antonio Salgueiro Aquino.
- Birth date: august 5th 1970
- Education: B.Sc. in Fundamental Physics, University of Santiago de Compostela University 1988-1993.
- Address: Marin (Pontevedra). Spain.
- Current position: Secondary School teacher in Technology.

Working times (planned) : 300 hours. From 6th August, to 30 September, on an eight hours day basis.

Technologies involved:

- Java ™language.
- Android ™. The operatig skeleton application.
- PhoneGap ™(alias Cordova) framework to develop multi-platform applications.
- JQuery and JqueryMobile to development of mobile oriented applications.
- JavaScript with Web Databases.
- Git for version control system.
- LaTeX for documentation.

Meetings:

- Technologies to be used were stated, work methodologies, first application windows (pages), Android version to be used (2.3.3) because is the most popular.
- Several emails and gtalk conversations about organization, general problems were written.

Teleworking is carried on

Materials and special equipment used:

- Hardware: Intel Quad, 6GiB RAM, 500GiB HD.
- Software: Debian GNU/Linux Wheezy (testing), Eclipse Juno, JQuery 1.8.1, jQueryMobile 1.1.1, and PhoneGap-Cordova 1.8.1, Android Virtual Manager 2.3.3, Git 1.7.10.4-1.
- Real testing: Sony-Ericsson Xperia V mobile phone, with USB cable and wifi.

Contents

Copyright **i**

Acknowledgments **iii**

Description of the practicum **v**

1 Introduction **1**

2 Working plan **3**

2.1 Description and objectives 3

2.2 Motivation 7

2.3 Methodology 7

2.4 Tools 9

2.5 Work plan 10

2.5.1 HTML 10

2.5.2 Interface code 12

2.5.3 Database code 14

2.5.4 Database 16

2.5.5 Interface Diagram 17

3 State-of-art solutions **19**

3.1 Siestta 19

3.2 Sixa 22

3.3 Android applications 22

3.3.1 Grade Book 22

3.3.2 Attendance 24

3.3.3 Teacher Organizer 25

3.3.4 Teacher Aide 27

4 Description **29**
4.1 Preparation of development 29
4.2 Application Skeleton 30
4.3 Upload into Git repository 30
4.4 Development 31
4.4.1 Groups 32
4.4.2 Students 35
4.4.3 Attendance 35
4.4.4 Activities 37
4.4.5 Assessment 37
4.5 Database structure 37
4.6 Los algoritmos para el desarrollo de la solución 38
4.7 qué quieres resolver 38
4.8 cómo lo vas a hacer 38
4.9 herramientas conceptuales necesarias 38

5 Results **39**
5.1 Objectives completed 39
5.2 Technical details 39
5.3 Further objectives 39
5.4 Resultados en forma de tablas, gráficas e imágenes donde se describa cuantitativa y cualitativamente el funcionamiento de la aplicación 40
5.5 Análisis crítico de los resultados con el objetivo de decidir si el sistema implementado es válido 40

6 Conclusiones y trabajos futuros (5%) **43**
6.1 Implementación 43
6.2 Resultados 43

Bibliography **46**

List of Figures

2.1 Diagram 18

3.1 Siestta Attendance Page 19
3.2 Siestta Main Page 20
3.3 Siestta SQL 21
3.4 Sixa Main Page 22
3.5 Sixa Schedule Page 23
3.6 Sixa Reports Page 23
3.7 Grade Book 24
3.8 Attendance 25
3.9 Teacher Organizer 26
3.10 Teacher Aide 28

5.1 Siestta Main Page 40

Chapter 1

Introduction

In a high school, there are classes which attendance, assessment and group dynamics is difficult to control. This is especially relevant for the technology workshop, this could be a noisy, annoying, even dangerous place, which requires teacher's standing monmitoring. This workshop needs a lightweight, reliable and also complete software tool. A simple web solution is not enough, because it could be too complicated and not fast enough; even use a tablet can be heavy. Therefore, this problem requires a new approach.

The development of a multiplatform tool, open source, for smartphones, including tablets, for attendance monitoring and evaluation of students is suggested. To achieve this goal a client-side application will be developed using a multiplataform framework: Phonegap (Cordova) [Pho12c]. Phonegap allows you to develop quickly, utilizing well-known languages as Javascript and HTML. It permit us to deploy an application for both Android, WebOS, iOS and others.

For data management, a built-in database, SQLite, will be employed which has all capabilities needed (autoincremental indexes, relation among tables and several built-in functions).

The tool that enables rapid development of the application and integration of performance tests is Eclipse. Also Eclipse is well integrated with Android SDK, and its Android virtual machine (AVM).

Chapter 2

Working plan

2.1 Description and objectives

An open source multi-platform management application for high school teachers is developed. It can be run on a smartphone or tablet. The actual objectives of the application are students management:

- Attendance and punctuality control.
- Misbehaviour control.
- Activities assessment. Each student will have an activity mark.

Application would also include these features:

- Data visualization. As table-like format. Attendance and misbehaviour
- Server synchronization with a custom application or Xade [dEXdG12].

The final goal is to develop an application to make teacher's work easier and comfortable. This application is focused on user (teacher) experience. Also an objective is to write extensible, easy to read code, which allows external developers to take part into application development.

Below, a list of tasks to be done to fullfil application development:

- Study state-of-art solutions:

  Find out other solutions: PDAs and smart-phone or tablet related and web-based applications.

  Download to study and reuse graphical user interfaces, code or/and database structure.

- Develop database structure: tables (field names and type of data), and relationships among tables.

- Preparation for development:

  Build development and staging environment:

  Install Eclipse [Fou01], Android Virtual Machine [Inc01], Aptana Plugin for Eclipse, JQuery, JQueryMobile [Mob12a] and Phonegap [Pho12c] from their respective websites [Pho12b].

  Choose application name and folder's policy.

  Make a simple application: only a blank page.

  Configure a git repository and upload the application: [Aqu12].

Development will follow these stages:

- Populate database with sample data. Firstly, data will be hard-coded into source code to stagging. Several tables will be created: groups, students, sessions, teacher schedule, students attendance, activities, student activities, and activities per group. Secondly the appropiate windows to manage these tables will be built.

- Groups: Several groups (four) of students will be hard-coded into javascript source code, with three or four students each other.

  Make list of groups window. This will list the four groups.

  Groups management window. Another group could be added, or removed.

- Students information:

  Make list of students window per group and complete list of students.

  Students management window: to insert and update data students: name, surname, birthday, address, e-mail, tutor name, landline and cell phone numbers and nationality.

- Sessions. Each lecture has a description (as 'first hour', or 'recreation'), starting and ending time. These sessions will be hard-coded on first version.

- Teacher schedule. For the current teacher, it contains weekly and daily schedule: name of group, session and day of the week. This information will be also hard-coded.

- Main window (or *page* in PhoneGap notation) will be created. Teacher can set current data and go to *daily page*, manage activities, students and groups or manage reports: assessment and attendance. Next page will be *daily page*.
- Timetable for current date: daily schedule, list of groups for each day ordered by session.
- Attendance page will be next to be build. It contains a list of students by group. Teacher can assign an attendance code to each student (attendance, misbehaviour, unpunctuality or excused).
- Assessment page is next to previous one. It contains one upper list of activities and a list of students similar to *attendance page* . Teacher could set activity and assign adequate mark to each student.
- Activities. This page manage activities (add and disable activities [1]): name and description of assessable exercises.

  Activities group will be set in activities page, and activities student will be related with assessment.
- Reports pages.

  List activities.

  List groups.

  List students by group.

  List students attendance by week. User can select previous or next week, or set another date.

  List students marks and final mark.
- Error handling. While developing each SQL query, any error will be catched and an error window will appears.
- Eventually, test into real hardware: an Android 2.3.3 mobile phone, tablets, and so on.

Next steps will include :

- Load data from an external file. This is well-documented [Pho12d].
- Load images from disk (SD-Card).

[1] still not coded

- Save or download data from database to disk.

- Xade web interface.

  a) Study Xade web interface.

  b) Develop an ad-hoc application for retrieve Xade's data. With javascript or a native one.

  8. Develop an ad-hoc application for store data.

  9. Synchronization with a custom server or with Xade.

- Test units. To ensure previous step were implemented.

- User documentation. Manual with images.

- Developer's documentation.

- Find out a website to host a forum, a bug report system, documentation and application download.

Task above could be achieved in a 300 hour basis. In the following table a broad estimation of time spent in each task are shown.

Task	Time (hours)
State-of-art solutions	10
Develop database	8
Preparation for development	40
Development.	
Populate database with sample data.	20
Groups. List and management	50
Students. List and management.	30
Timetable for current date.	80
Add attendance, behaviour.	50
Add error handling.	2
Retrieve and insert data from and to database.	30
List of attendance, misbehaviour incidents.	12
Add activities grades for each student.	12
List students marks and final mark.	14
Activities management window.	14
Management of student notes.	20
List of student notes.	2
Test into real hardware	20
Total	384

2.2 Motivation

As a Technologies teacher, in my daily work I have to evaluate students work such as working with tools, cooperative work with other classmates etc., besides usual activities as written exercises. A long data sheet could used, or an awkward long spreadsheet, but a portable device with a custom application should be desirable.

This application tries to increase teacher's productivity because teacher only has to write attendance, or unpunctuality two times (on official report and on application's window), and classroom notes and activity grades on very easy way.

The most important feature is to be as easy, fast and intuitive as possible. It could be desirable to be platform independent (Android, iOS, Windows RT), but Android is preferred because it is open source and has a higher market share.

On the other hand, development of this application improves my computer science skills in mobile-phone applications development: JQuery, jQueryMobile, PhoneGap/Cordova, SQLite, Android, git repository management.

2.3 Methodology

This work was carried on building small blocks, also called pages, and make it up into final application. A page is a visible window, only first page is visible, and other pages are called from this one.

Database structure was separated from interface, and interface was also separated into dynamic and static contents. Each new functionality was written, tested, and polished.

Foremost a new window/page is designed from previous HTML code or from sample code [Cas12] and written in **index.html** , *id* are set with ad-hoc names (e.g. *id="edit_student"*, *id="in_name_student"*), these *id's* are used in javascript code.

Sample Page. index.html

```
<div data-role="page" id="daily\_work">
    <div data-role="header" data-add-back-btn="true" data-theme="a">
                <h1>Day</h1>
    </div>
    <div data-role="content" data-theme="a">
        <div class="ui-grid-b">
            <div class="ui-block-a">
                <div data-role="fieldcontain">
                    <label for="date"  >Date:</label>
                </div>
            </div>
            (...)
        </div>
    </div>
    <div data-role="navbar">
        <ul>
            <li>
                <a href="#" onClick="onGeneralFile()"
                data-role="button" data-icon="star"
                data-theme="a" > File</a>
            </li>
            <li>
            (...)
            </li>
        </ul>
    </div><!-- /navbar -->
    <div data-role="footer" class="footer-docs" data-theme="a">
        <p   style="text-align:center;"  id="teachers\_name"></p>
    </div>
</div>
```

Secondly **interface.js** code are written: show progress icon (a *PhoneGap* function), a customized function which loads data into new HTML code (e.g. *loadGroupAssessment*) , and, finally a *PhoneGap* function to show new page/window.

Sample interface.js code

```
$.mobile.showPageLoadingMsg();
id_group=global_id_group;
loadGroupAssessment(global_db,id_group);
$.mobile.changePage("#list_students_assessment");
```

Thirdly **database.js** code are written: Declaration of previous function **??**, which usually implements a SQL-query to SQLite database. These functions also fill html code, so HTML *id*'s names are so significant.

Sample database.js code

```
var list_asset=$('#list_assessment_select');
sql="SELECT ";
sql += " ACTIVITIES_GROUP.a_date (...)  ";

db.transaction(function(tx) {
    tx.executeSql(sql,[],
        dbSuccessFunc = function(tx, results) {
           var html ="";
           for (var i=0;i<results.rows.length;i++) {
              a_id = results.rows.item(i).a_id;
              a_name = results.rows.item(i).a_name;
                    (...)
              html +=' <option value="'+a_id+'">'+a_name+'</option> ';
           }
           list_asset.empty().append(html);
           list_asset[0].selectedIndex = 0;

           if(results.rows.length>0) {
             $('#current_group_assessment').text(
                results.rows.item(0).g_data);
           }
           global_id_group = id_group;
           listStudentsByGroupAssessment( db, id_group,
            $('#students_assessment_ul'));
       },
       dbErrorFunc (...)
        );
});
```

After successful tests using Android Emulator provided by Android SDK [Inc01], **TODO** file could be updated, and code will be upload to remote git [Aqu12] repository.

Git init shell

```
$ git add -A *
$ git commit -m BRIEF_DESCRIPTION_OF_CHANGES
$ git push origin master
```

From time to time application is downloaded from git repository into real hardware and tested. It is usually faster on a mobile-phone than on emulator.

2.4 Tools

Tools involved in *EduXes* development are :

- IDE. Eclipse [Fou01]. The most popular Integrated Development Environment for Java and other languages. It integrates browser, contextual help, even Android Virtual Machine is integrated.
- Aptana [App05]. Aptana Plugin is used HTML, CSS and JavaScript editing.
- Android Development Kit [Inc01]. Android SDK provides the API libraries and developer tools necessary to build, test, and debug apps for Android.
- Git. Also included as an Eclipse plugin.

To build this documentation several applications were used:

- Sqlfairy. Tranforms SQL language into a png image.
- Gimp 2.8.2 to get screen-shots.
- LibreOffice 3.5.4.2 to write previous document
- LateXila 2.7.0 pre to write this document.
- Other tools from GNU/Debian Wheezy October 2012

2.5 Work plan

Following previous methodology 2.3, a detailed plan is shown below:

- HTML. User interface.
- interface.js. Functional code and database agnostic.
- database.js. Database related code.
- DATABASE.sql. Database declarations as a standalone file.

2.5.1 HTML

There are only two html files: **index.html** and **remove.html**. The most important: **index.html** file is build by blocks called pages, gathered together [Mob12b]. Each page is its own **div** with custom properties (*propieraty* properties in Eclipse jargon) : **data-role="page"**. Below as show an example: list of several reports, user could choose one and a function (*onReportListAttendance()* or *onReportListAssessment()*) is executed .

Inside each page there are several identificatives *id="name"* which are used by application to fill with data (e.g. within *daily_schedule* page *current_day* id is used to set date to current date).

Reports Page

```
<div data-role="page" id="list_reports" data-add-back-btn="false">
            <div data-role="header"  data-back-btn-text="previous"
              data-add-back-btn="false" >
            <a href="#daily_work"  data-icon="arrow-l" data-theme="a"
              data-role="button">Back</a>
                <h1>Reports</h1>
            </div>
            <div data-role="content">

                <ul data-role="listview" id="list_reports_ul"
                  data-split-icon="gear" data-split-theme="a"
                  data-filter="false"
                  data-inset="true" data-theme="a" >
                <li><a href="#" onClick="onReportListAttendance();"
                 >Attendance</a></li>
                <li><a href="#" onClick="onReportListAssessment();"
                >Assessment</a></li>
                <li><a href="#" onClick="" >File</a></li>
                </ul>
            </div>
            <div data-role="footer" class="footer-docs"
                data-rel="back"
             data-theme="a">
             </div>
</div>
```

A complete list of pages is shown below:

List of Pages

intro	Introductiony page
daily_work	daily work
list_reports	List reports
list_settings	List settings
file	Exit application
daily_schedule	List of groups per day
list_students_attendance	
list_students_assessment	
list_students_attendance_by_group	
general_list_attendance	
list_groups_attendance	
list_groups_assessment	
list_students_assessment_reports	
list_groupsto_edit	
edit_new_group	
edit_groups	
list_groups	
list_students_by_group	
list_students	
edit_student	
edit_new_student	
list_all_activities	
update_activity	

2.5.2 Interface code

There is only onle file which deals with interface interactions (events from **html** files): *interface.js*. If the function is called directly from *html* code, function name contains **on** prefix. There are several groups of functions:

- *Students* and *Groups* functions, which manage and list general students and groups information.
- *Activities* functions, which update, list and add new activities.
- *Assessment* functions, list and update students marks.
- *Attendance* functions, list when student attend classes and update their values.

Whether these functions need access to data (every function but *onGeneral**) they call their counterpart function in **database.js** file.

List of Functions: interface.js

Name	Task	Group
onDeviceReady()	Firstly loaded, create and populate database, load initial page	Initial
init()	Initial	
initialize_ data ()	Load default date	Initial
open_ daily_ page()	Daily work page: list of groups	Schedule
listStudentsAttendance()	Students Attendance page	
onAddNewGroup()	Edit Group	Groups
onAddNewGroup()	Update Group	Groups
onDeleteGroup()	Delete Group	Groups
onSaveNewGroup()	Save Group	Groups
onListAllGroups()	List All Groups	Groups
EditStudent()	Edit Student	Students
onDeleteStudent()	Delete Student	Students
onAddNewStudent()	Add new Student	Students
onSaveStudent()	Save Student	Students
onSaveNewStudent()	Save New Student	Students
listStudents()	Full list of students	Students
listStudentsByGroup( )	List Students by group	Students
onListAllStudents()	Full list of students	Students
onListAllActivities()	List All Activities	Activities
onUpdateActivity()	Activity update	Activities
onAddNewActivity()	Add new Activity	Activities
onSaveNewActivity()	Save New Activity	Activities
onOpenStudentsAssessment()	Open Students Assessment	Assessment
onRefreshGroupAssessment()	Reload Group Assessment	Assessment
onOpenStudentsAttendance()	Open Student Attendance	Attendance
onReportListAttendance()	List Attendance	Attendance Reports
listStudentsByGroupAttendance()	List Students by Group Attendance	Attendance Reports
studentsAttendanceListPrevious()	List Students by Group Previous week	Attendance Reports
studentsAttendanceListNext()	List Students by Group Next week	Attendance Reports
onReportListAssessment()	List Groups for Assessment	Assessment Reports
onListStudentsAssessment()	List Students for Assessment	Assessment Reports
studentState()	Student State	Student
Attendance()	Student Attendance	Student Attendance
onGeneralFile()	Exit	Exit
onGeneralListReports()	List Reports	Reports
onGeneralListSettings()	List Settings	Settings

2.5.3 Database code

The file which contains functions which carry on querys and data manipulation (on data base (SQLite)) is : *database.js*

There are global variables on top of this file. These variables are used by functions because there are not obvious ways to pass values from **html** code through functions (e.g. as day week, database, current date, ...):

Variable	Meaning
`global_id`	General and unique identification of any table as students or groups[a]
`table_global`	students, groups table
`global_id_group`	Identification of a group
`global_id_student`	Identification of a student
`global_id_activity`	Identification of an activity
`global_max_activities`	Number of activities
`global_no_groups`	Number of groups
`global_week_day`	Number of week day (0-6)
`global_db`	Database pointer
`global_session`	Selected session
`global_actual_date`	Current date
`global_reports_date`	Date for reports
`global_is_new`	Whether activity is new
`global_exist`	if exist current record
`STATE_NONE`	Default Student state
`STATE_ABSENCE`	Truancy
`STATE_UNPUNCTUAL`	Unpunctuality
`STATE_EXCUSED`	Excused unattendance
`STATE_BEHAVIOR`	Bad behaviour

[a] will be deprecated

Several approaches have been considered, a lot of tiny, one function equals one simple task, but code were growing in complexity, became very difficult to read, maintain and pass variables to them. Other perspective was to write only several complete functions which carry all (or almost all) the work. From this point of view functions become longer, (up to a hundred of lines of code) but easier to test and follow.

Next tables list every function used in **database.js**, gathered together by task, or menu options.

List of Functions: database.js

Name	Task
errorCB()	Error handler
successCB()	Success handler
⌜ *loadSchedule()*	Main Window → Load Daily Schedule
querySchedulePerDayDB()	
⌞ *queryScheduleSuccess()*	
⌜ *loadAllStudents()*	Settings →List all students
queryAllStudentsDB()	
⌞ *queryAllStudentsSuccess()*	
⌜ *loadStudent()*	Settings →List all students → Fill Students page
queryStudentSuccess()	
queryStudentDB()	
⌜ *loadNewStudent()*	Settings → List all students → New Student
⌞ *queryNewStudentDB()*	
⌜ *loadStudents()*	Delete Student →List all students
queryStudentsDB()	
⌞ *queryStudentsSuccess()*	
loadStudentsByGroup()	Settings → Groups → List Students
⌜ *loadStudentAttendance()*	Main Window → Load Daily Schedule → Groups (Default: Attendance)
queryStudentsAttendanceDB()	
queryStudentsAttendanceSuccess()	
⌞ *fillSelectStudent()*	
⌈ *loadAllActivities()*	Settings → Activities
⌞ *queryLoadAllActivitiesDB()*	
loadActivity()	Settings → Activities List → Load activity information
updateActivity()	Settings → Activities List → Load activity → Update Activity.
updateActivitiesGroup()	idem. for each group
insertNewActivity()	Settings -> Activities -> New
insertActivitiesGroup()	idem. for each group
loadGroupsActivitiesEdit( )	Settings → Activities → New (List groups)
⌜ *listMaxActivities( )*	List Number of activities
⌞ *queryListMaxActivities()*	
Peers ()	Activities auxiliar object

List of Functions (2): database.js

Name	Task
insertNewGroup()	Settings → Groups → Insert Group
updateGroup()	Settings → Groups → Update Group
deleteGroup()	Settings → Groups → Delete Group
loadGroup()	Settings → Groups → Select
⌜ *loadAllGroups()*	Settings → Groups
⌞ *queryAllGroupsDB()*	
insertNewStudent()	Settings → Insert new Student
⌜ *querySaveStudentDB()*	Settings → Update Student
⌞ *saveStudent()*	
deleteStudent()	Settings → Delete Student
insertStudentStateL( )	Settings → Insert New Student
updateStudentStateL( )	Settings → Update Student
⌜ *loadGroupAssessment( )*	Assessment [a] → Group → List options
listStudentsByGroupAssessment()	
refreshGroupAssessment()	
⌞ *fillSelectStudentAssessment( )*	
onChangeStudentAssessment()	Assessment → Change Student Mark
updateStudentAssessmentL()	Id. Update Student Assessment
insertStudentAssessmentL()	Id. Insert Student Assessment
loadGroupsAssessment()	Report → Assessment → Group
loadStudentsAssessment()	Report → Assessment → Students [b]
⌜ *loadGroupsAttendance()*	Reports → Attendance → List groups
⌞ *queryGroupsAttendanceDB()*	
⌜ *reportAttendanceDB()*	Reports → Attendance → Group
queryReportAttendanceDB()	
⌞ *queryReportAttendanceSuccess()*	
deleteRawRecord( )	Delete any row from any table
stateCheck( )	Check whether student state changes
updateStudentState( )	Update student state

[a] Main Window → Attendance → go to Assessment
[b] Assessment List

2.5.4 Database

Database is the application most important data struct, it requires a special study. Actual database is inherit of Siestta application [P06]. This database was developed from *groups* table, to *activities_student* following less related path (*groups* → *students* → *session* → *teacher_schedule* → *attendance* . . .).

Snippet database

```
(...)
-- Students Attendance
     DROP TABLE IF EXISTS ATTENDANCE;
     CREATE TABLE IF NOT EXISTS ATTENDANCE (id integer primary key ,
      id_group integer, id_student integer, id_session integer,
        a_type integer, a_date text,
     FOREIGN KEY (id_student) REFERENCES STUDENTS (id),
     FOREIGN KEY (id_group) REFERENCES GROUPS(id),
     FOREIGN KEY (id_session) REFERENCES SESSIONS(id) );

-- Activities
     DROP TABLE IF EXISTS ACTIVITIES ;
     CREATE TABLE IF NOT EXISTS ACTIVITIES
             (id integer primary key, name text, date_init text,
             date_end text, weight integer, final integer );
     DROP TABLE IF EXISTS activities_student;
     (...)
```

2.5.5 Interface Diagram

This figure 2.1 ilustrates application workflow. Dotted lines are instructions and continuous lines data flow. Readings from database are not displayed.

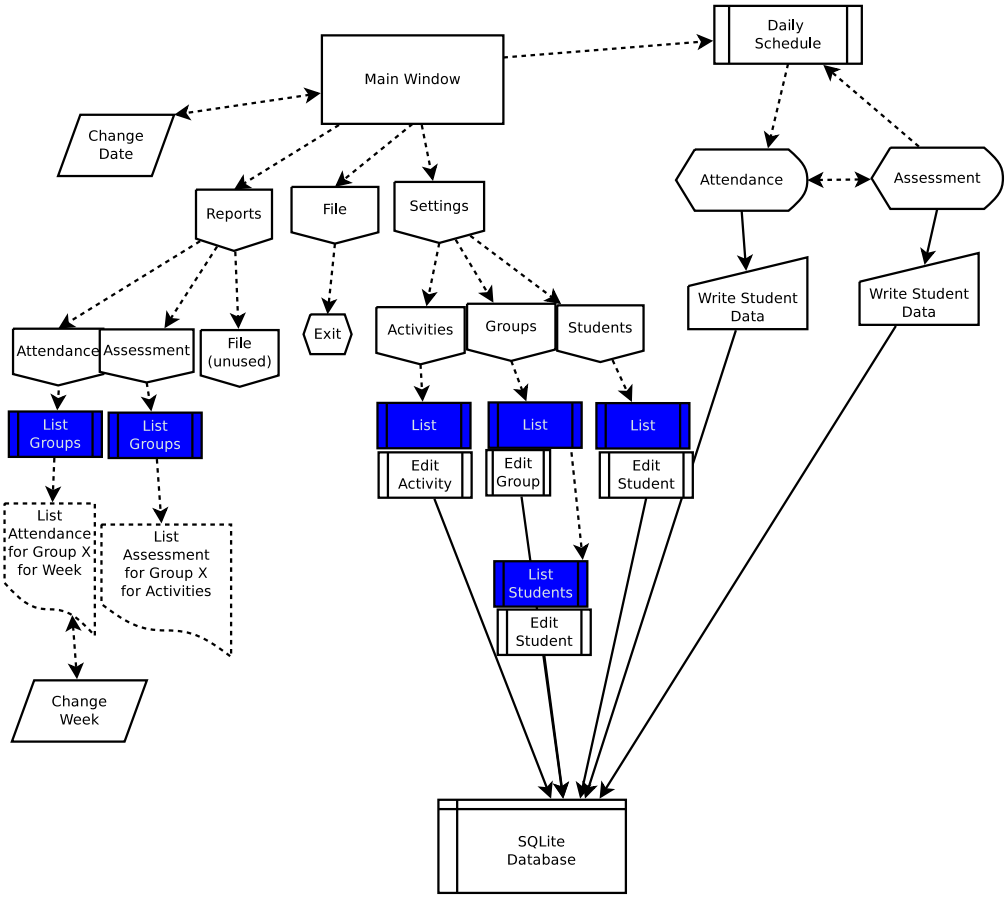

Figure 2.1: Diagram

Chapter 3

State-of-art solutions

Only one open source application was found suitable for study, Siestta , nevertheless there are a lot of educational software (Sixa [S.L09], Unisoft [Uni05]) but they are privative, Microsoft Windows freeware or both (SAS académico).

3.1 Siestta

Technically Siestta[PÓ6] is an GPL'ed 1990's style PHP-based web application with Ajax, an interactive editor, fckeditor and fpdf to generate reports 3.1.

From user point-of-view there are online documentation. This application includes management of students, attendance, marks, tasks, incidents, general queries, letters to parents, interviews with parents, messages, appointments, exams and more.

Several screen-shots were taken and its structure will be reused in current application 3.2:

This application are also available for PDAs, it could be a valid solution but it is server-side with outdated technologies. Data structure from Siestta is standard and fully functional, and it will be partially reused by EduXes.3.3.

Source code are also shown: calendario.php. It shows us a PHP application which uses sessions variables and is not Model-View-Controller oriented.

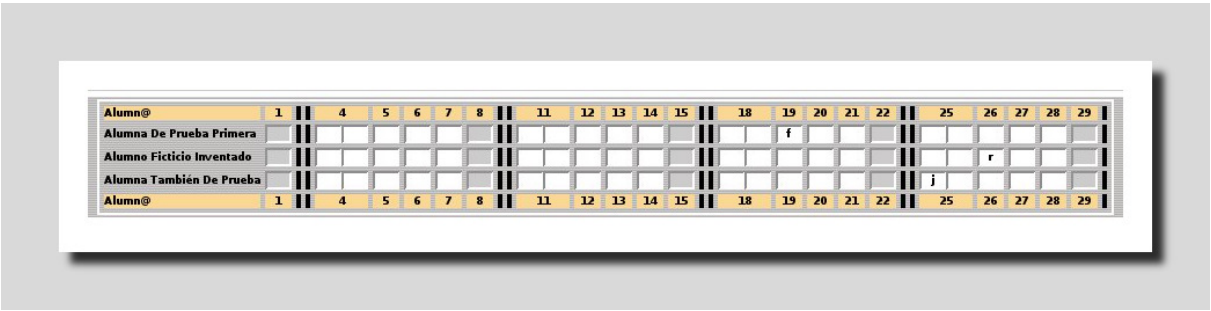

Figure 3.1: Siestta Attendance Page

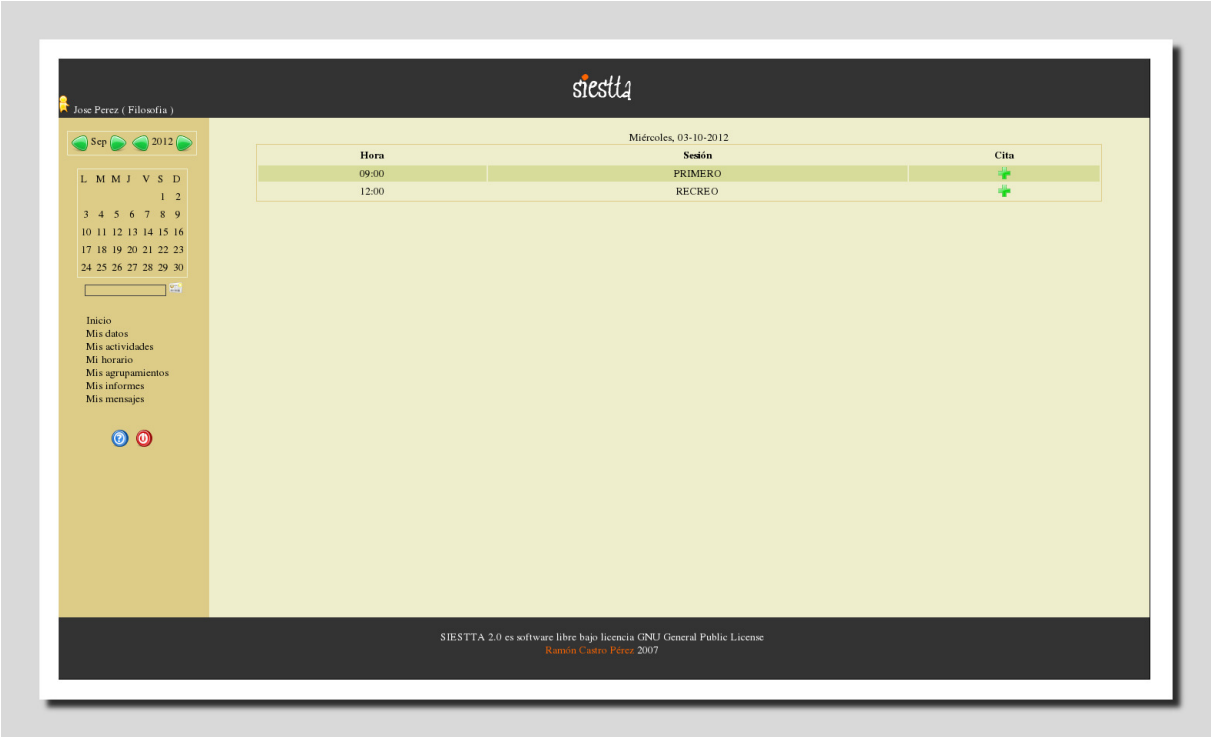

Figure 3.2: Siestta Main Page

Sample Siestta code: calendario.php

```
<?php
session_start();
require('config.php');
require('idioma/'.$idioma.'');
include('funciones_calendario.php');
$docente = $_SESSION['usuario_sesion'];
//recogemos variables
$mes_actual = $_POST['mes'];
$anyo_actual = $_POST['anyo'];
if($mes_actual || $anyo_actual) {
include('funciones.php');
conecta();
}
//si es la primera vez que entramos, cargamos la fecha actual
if(!isset($mes_actual)) $mes_actual = date('m');
if(!isset($anyo_actual)) $anyo_actual = date('Y');
//presentamos ahora el calendario del mes actual o cargado
//tabla con nombre mes y año y las flechas para navegar
echo '
<br />
<table class="tablacentrada_i">
<tr>
<td>
<a href="#" onclick="navegaMes(\''.$mes_actual
.'\',\''.$anyo_actual.'\',\'menos\')" title="'.$id_anterior
.'"><img src="imgs/anterior_peq.png" class="alin_bajo" alt="'
.$id_anterior.'" /></a>
';
```

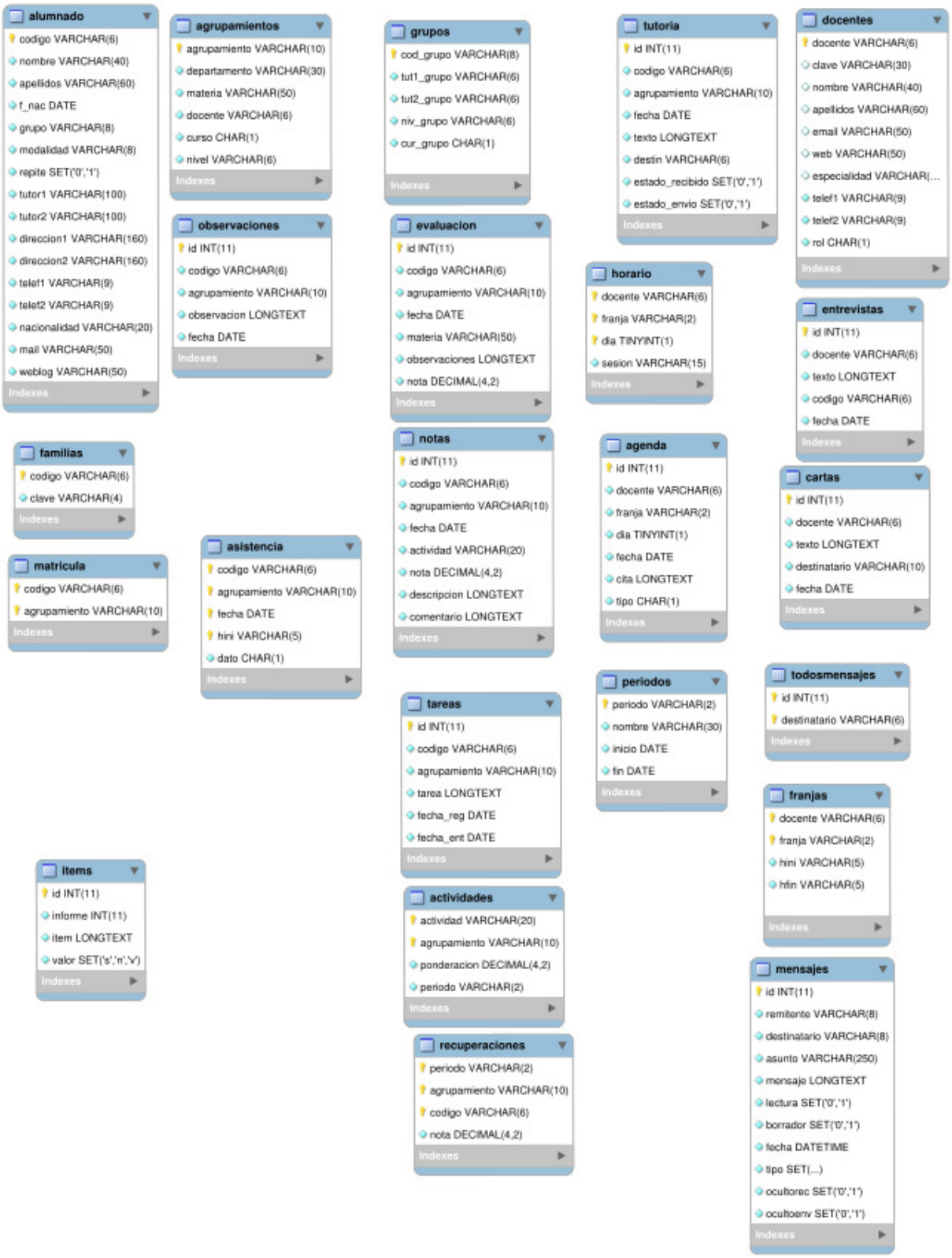

Figure 3.3: Siestta SQL

Figure 3.4: Sixa Main Page

3.2 Sixa

Although not GPL'ed, and there are no source code available, several design ideas could be considered: for main page 3.4, schedule page 3.5 and reports page 3.6.

It has a PDA version (not shown), and it Xade data import capable. Also has a long list of features [S.L09].

3.3 Android applications

Several Android applications are listed below, in a short list:

3.3.1 Grade Book

- Name: Grade Book [Aca11]

Figure 3.5: Sixa Schedule Page

Figure 3.6: Sixa Reports Page

Figure 3.7: Grade Book

- Description:Now teachers can manage their students grades directly on their Android device!
- Key Features:

  No need to sync two separate grade books! Use one primary grade book on Google Spreadsheets

  Email student grades with the click of a button

  Pin-number to protect your grade books if your phone is lost
- Updated:July 7, 2011.
- Price :4 €
- Screenshot: 3.7

3.3.2 Attendance

- Name: Attendance [fA10]
- Description:A great way to take attendance. Put your student list in a Google Spreadsheet, and this app takes care of the rest. There is no need to enter total

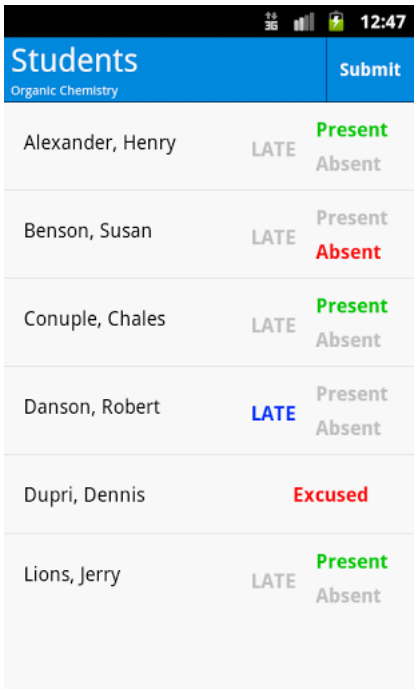

Figure 3.8: Attendance

absences into a spreadsheet at the end of the semester because all absences/tardies are calculated in your google spreadsheet each day you take attendance.

- Key Features: attendance.
- Updated:January 13, 2012.
- Price : Free
- Screenshot: **??**

3.3.3 Teacher Organizer

- Name: Teacher Organizer [Lle12]
- Description:Gradebook and attendance, notes, schedule a teacher (high school teacher).
- Key Features: Unified information resource teacher developed within diploma projects.
  - Functional log attendance and log evaluations point system of evaluation, while allowing classes to celebrate and stand missing the mark;

    Compilation of the magazine is "on the fly", depending on the selected students;

Figure 3.9: Teacher Organizer

Selection of students by selecting a training group, not the individual student;

Log open it is possible to select more than one group of students;

– Entering information about your child and watching it like when viewing the log evaluations or visits, and without the need to open the log;

– Notebook;

Notebook allows you to store notes, view, and edit;

A system for cataloging notes for quick search;

The possibility of entering one note in more than one subject group;

– Schedule a private teacher;

It is possible to view the schedule for the next semester as up to now, and after;

The system can generate the schedule for a few weeks in advance, if it is repeated;

Supports two-week schedule (odd and even weeks).

- Updated: December 26, 2012.
- Price : Free.
- Screenshot: 3.9

It has a drawback, its web page is in Russian, and do not translated. This application seems to be very professional. There is no code avaliable.

3.3.4 Teacher Aide

- Name: Teacher Aide [iagdc12].
- Description:The app allows teachers to take attendance and record grades on their phone or tablet.
- Key Features:
  - default values set to Present for fast attendance
  - import student names via CSV file
  - export data via CSV generated file send via email or to Dropbox.
  - 1-click text to students/parents for tardy/absent students and missing assignments.
  - 1-click Random student generator (no more popsicle sticks)
  - Grading interface to allow recording of assignments using Yes/No/Missing, Points, Letter Grades, Percent.
  - Generate PDF file for attendance and grade reports.
  - Print reports directly to Google Cloud Printer from app.
  - Allows user to send bulk text message to parents and students
  - Call parents/student right from the App
  - Send bulk email to parents/students from app without having to confirm 1 at a time
  - Allows the user to save or load student/parent contacts from the device contact list.
- Updated: 26 December 2012.
- Price : Free (Limited number of students).
- Screenshot: 3.10

Excellent but not open source or even free.

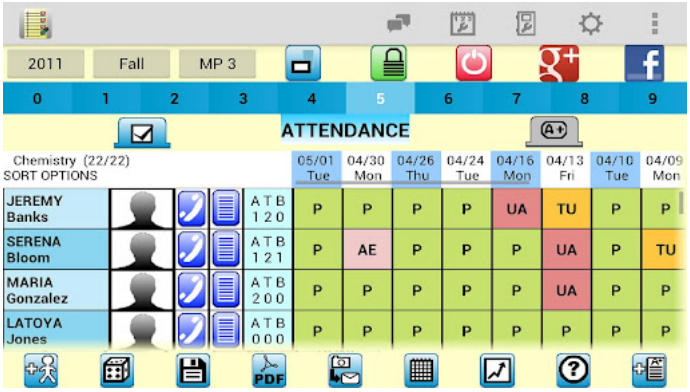

Figure 3.10: Teacher Aide

Chapter 4

Description

Below a detailed list of procedures to build and development itself *EduXes* application, will be described.

4.1 Preparation of development

Firstly Java Development Kit (JDK) version 1.6 is needed, to build application itself and the Eclipse IDE, is downloaded from *Oracle*: [Ora95]

`http://www.oracle.com/technetwork/java/javase/downloads/index.html`

As root user that file is unpacked into **/usr/lib/jvm** and configured to be the Java default:

Update Java

```
# update-java-alternatives -s JDK_1.6_NAME
```

Eclipse Juno (4.6) is downloaded from its site [Fou01]: *Download* → *Linux* 64 *bits*

`http://www.eclipse.org`

Android Development Toolkit (ADT)[Inc01] is downloaded following instructions on this page:

`http://developer.android.com/sdk/installing/installing-adt.html`

In Eclipse a new line is included into repository software (*Help* → *Install New Software* → *Add*):

`http://dl-ssl.google.com/android/eclipse/`

Next step is to select all the related software listed.

For Aptana Plugin the line to be added into Eclipse is:

`http://download.aptana.com/studio3/plugin/install`

Furthermore JQuery, JQueryMobile and Phonegap are needed, and were downloaded from their web sites:

- JQuery 1.8.1 (no newer versions), from `http://jquery.com/` :

  JQuery will be copied into **assets/www/js** folder.

- JQueryMobile 1.1.1 from `http://jquerymobile.com/`:

  JQueryMobile is a zip file which will be uncompressed and copied into **assets/www/js** folder.

- PhoneGap - Cordova 1.8.1 is downloaded from `https://github.com/phonegap/phonegap/zipball/1.8.1` and installed following reference [Pho12b]

To create a PhoneGap application there are very important instructions (Getting Started with Android) should be followed step by step: [Pho12a]

4.2 Application Skeleton

Next step is to choose application name and folders policy. Name *EduXes* stands for "Educaciń" and "Xestión", is a educational management software. About folders policy, a folder is created (**assets/www/js**) which contains javascript (***.js**) files except *JQuery* and *JQueryMobile* which is included into another folder (**assets/www/js/jquery**), do not forget style sheets files (***.css**)

To finish building application skeleton a simple application is created which contains only a blank page.

Getting started with Android [Pho12a] is followed step by step.

4.3 Upload into Git repository

Following stage is to upload that simple application into a git repository. An account is created in Github[Git08], which is the largest code host in the world, and a new application is initialized. This are the source code project page `https://github.com/joseantoniosa/EduXes`.

Then source code are upload to Github:

Git init shell

```
$ git init
$ git add -A *
$ git remote add  EduXes git@github.com:joseantoniosa/EduXes.git
$ git push origin master
```

Each time an update in code is done, code is uploaded :

Git update

```
$ git add -A *
$ git commit -m 'CHANGES_DESCRIPTION'
$ git push origin master
```

4.4 Development

Before begining with development itself is compulsory to explain some concepts already shown above in Methodology (2.3), but with more detail: JQueryMobile [Mob12a] and by extension, PhoneGap [Pho12c] applications, are structured in pages:

which are very similar to desktop applications windows. Therefore, from Javascript code, to change to a new page

$.mobile.changePage("#daily_work") opens daily_work page.

As shown before (2.3) *EduXes* contains several files:

- Four files contain application code itself:

  interface.js : It contains information and decisions related to interface and application workflow, completely independent from database. It is located in **assets/www/js** folder.

  database.js : It contains database related code: SELECT, INSERT, etc. It is located in **assets/www/js** folder.

  index.html, **remove.html** only contain HTML framework, page properties, and static content. They are located in **assets/www/** folder.

- There are three important files which contain documentation:

  TODO.txt. List of goals to be achieved and milestone reached. It is located in root application folder.

DATABASE.sql. Data-base structure in SQL format. This file are shown below. It is located in root application folder.

REAME.txt. Only contains general information about this application.It is located in root application folder.

Also, *Eclipse* generates several files, the most important is the application file, which is an **apk** package: **bin/EduXes.apk**.

Next step is to build database, this will be described below 4.5. Following database, sample data is needed to begin application development .

4.4.1 Groups

The first window to be developed is *List of Groups*: as shown above 2.3, firstly a *page* is created inside **index.html**. This page, as every page, has three important elements:

- *header*. It could contain a backward button, which return application flow to previous visited page. This is disabled, because it could cause problems when previous page is an edit page, and another solution is preferred.

- *content*. Data itself, it has an important *id* to be filled with content by javascript code.

- *footer*. Usually page name.

List Groups page. index.html

```
<div data-role="page" id="list_groups" data-add-back-btn="true">
    <div data-role="header"  data-add-back-btn="false"  >
        <a href="#list_settings"  data-icon="arrow-l"
             data-theme="a" data-role="button">Back</a>
        <h1>Groups List</h1>
    </div>

    <div data-role="content">
    <ul id="groups_ul" data-role="listview" data-inset="true"
        data-split-icon="gear" data-split-theme="a" >
    </ul>
     <hr />
     <a  style="text-align: center" onClick="onAddNewGroup();"
        data-role="button" data-icon="add">Add New Group</a>
    </div>
    <div data-role="footer"  data-rel="back" data-theme="a">
     <p style="text-align: center">List of Groups</p>
    </div>
</div>
```

To fill this list of elements *<ul id=groups_ ul (...)>* a function *onListAllGroups()* is called. Firstly a `loading ...` logo appears, then a function from **database.js** is called (*loadAllGroups()*).

List Group page. interface.js

```
function onListAllGroups(){
    $.mobile.showPageLoadingMsg();
    table_global='GROUPS';
    loadAllGroups(global_db); // #groups_ul
    $.mobile.changePage("#list_groups", { transition: "slideup"} );
}
```

List Group page. database.js

```
function queryAllGroupsDB(tx) {
    log("Query All Groups \n");
    var ul_list =$('#groups_ul');
    tx.executeSql('SELECT * FROM GROUPS', [],
        dbSuccessFunc = function(tx, rs) {
        ul_list.empty();
        var html ="";
        for (var i = 0; i < rs.rows.length; i++) {
          id = rs.rows.item(i).id ;
          html = "<li>";
          html += "<a onClick='global_id=" + id +
              "; global_id_group="+id
              +"; table_global=\"groups\"; ";
          html += " listStudentsByGroup(" + id + " );'  ";
          html += " href='#' >";
          html += rs.rows.item(i).data;
          html += "</a>";
          html += " <a data-role='button' data-position-to='window'";
          html += " data-iconpos='notext' ";
          html += " style='float:right;' href='#' ";
          html += " data-rel='dialog' data-theme='a' ";
          html += " data-transition='slideup' ";
          html += " onClick=\"EditGroup(" + id + "); ";
          html += " global_id_group="+id
              +"; global_id=" + id + "; \">Edit</a>";
          html +="</li>";
          ul_list.append(html);
        }
        ul_list.listview('refresh'); },
        dbErrorFunc = function(ttx, e) {
            if (ttx.message)
                e = ttx;
            log(" There has been an error Select * from groups : "
            + e.message);
        return false;
    });
}
```

Second window to be developed is edit group information.

Edit Group page. index.html

```
<div data-role="page" id="edit_groups" data-add-back-btn="true">
    <div data-role="header"  data-add-back-btn="true" >
        <h1>Group Edition</h1>
    </div>
    <div data-role="content">
        <label for="nombre_grupo">Name</label>
        <input id="in_nombre_grupo" enabled="true" />
        <label for="nivel_grupo">Level</label>
        <input id="in_nivel_grupo" enabled="true" />
        <a onClick="onUpdateGroup();" data-role="button"
            data-icon="check">Save</a>
</div>
</div>
```

4.4.2 Students

4.4.3 Attendance

Main page. Repor Attendance.

4.4.4 Activities

4.4.5 Assessment

4.5 Database structure

EduXes Database structure

```
-- Groups
CREATE TABLE IF NOT EXISTS GROUPS (id  integer primary key ,
    data text , other_data text);
--- Students
CREATE TABLE IF NOT EXISTS STUDENTS (
      id integer primary key, id_group integer not null,
       name text, surname text,
      repeated integer, n_date text , photo text,
      tutor TEXT, address TEXT, phone text, e_phone text,
       nation text,
      FOREIGN KEY(id_group) REFERENCES GROUPS(id));
-- Sessions ( franja horaria)
CREATE TABLE IF NOT EXISTS SESSIONS (id  integer primary key,
        description text, h_start text, h_end text);
-- Teacher's schedule
CREATE TABLE IF NOT EXISTS TEACHER_SCHEDULE (id  integer primary key,
      id_session integer, day integer, id_group integer,
      FOREIGN KEY(id_group) REFERENCES GROUPS(id),
      FOREIGN KEY(id_session) REFERENCES SESSIONS(id));
-- Students Attendance
CREATE TABLE IF NOT EXISTS ATTENDANCE (id integer primary key ,
      id_group integer, id_student integer, id_session integer,
      a_type integer, a_date text,
      FOREIGN KEY (id_student) REFERENCES STUDENTS (id),
      FOREIGN KEY (id_group) REFERENCES GROUPS(id),
      FOREIGN KEY (id_session) REFERENCES SESSIONS(id) );
-- Activities
CREATE TABLE IF NOT EXISTS ACTIVITIES
    (id integer primary key, name text, date_init text,
    date_end text, weight integer, final integer );
CREATE TABLE IF NOT EXISTS activities_student
    (id integer primary key ,  id_student integer,
     id_activity integer,
    mark integer, a_date text, notes text,
    FOREIGN KEY (id_student) REFERENCES students (id),
    FOREIGN KEY (id_activity) REFERENCES activities(id) );
CREATE TABLE IF NOT EXISTS activities_group
    (id integer primary key ,  id_group integer,
     id_activity integer,
    enabled integer, a_date text, notes text,
    FOREIGN KEY (id_group) REFERENCES groups (id),
    FOREIGN KEY (id_activity) REFERENCES activities(id) );
```

4.6 Los algoritmos para el desarrollo de la solución

...

4.7 qué quieres resolver

...

4.8 cómo lo vas a hacer

...

4.9 herramientas conceptuales necesarias

...

NOISE vvvv

ASD

Chapter 5

Results

Application is evolving from list, edit students and groups, to its final goals. These objectives were fulfilled:

5.1 Objectives completed

- Access to any workday, any group and student.
- Management of attendance and misbehaviour of each student. The students information is still hard-coded into source files.
-

5.2 Technical details

Javascript-SQL conexion Asynchronous methods

5.3 Further objectives

- Links to student and group management. These pages were done but links are not missing into main application window.
- There are several objectives not fulfilled yet, but I am on the way to get those done, those are, in priority order:
- Data visualization. Student attendance and misbehaviour have to be shown in table-like window.
- Test into real hardware. EduXes.apk has to be copied into mobile phone.
- Activities evaluation per student. A window to display activities marks and final mark.

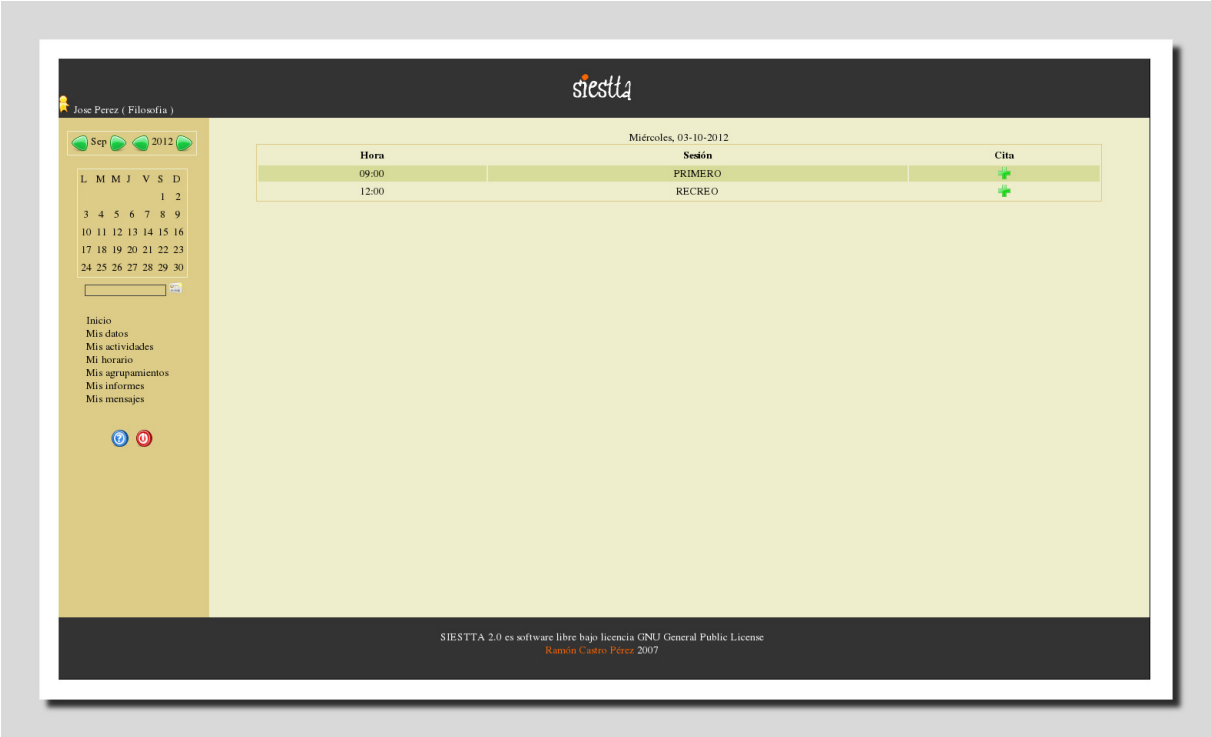

Figure 5.1: Siestta Main Page

- Timetable management. A window to manage groups timetable. When a group has class with this teacher.

- Server synchronization with a custom application or

These objectives were not fulfilled because time and skills lack.

En este apartado deberán quedar reflejados los experimentos realizados. Para ello se mostrarán: 5.1

5.4 Resultados en forma de tablas, gráficas e imágenes donde se describa cuantitativa y cualitativamente el funcionamiento de la aplicación

...

5.5 Análisis crítico de los resultados con el objetivo de decidir si el sistema implementado es válido

...

Several problems were faced: Eclipse environment: A stable, reliable and up-to-date IDE, with several plug-ins is needed. Download vanilla Eclipse Juno from its web-site is chosen because it is more stable, reliable, compatible with newer versions. Aptana Javascript plugin was chosen because Aptana allows source code auto-completion in JQuery. PhoneGap and Android incompatibilities. Android 2.3.3 requires JQuery-1.8.1 and does not work on higher versions. Error handlers. I have had several problems with tx.executeSql(...) function, it confused me with db.transaction(...): tx.executeSql(sql, [parameters], successHandler, errorHandler) and db.transaction(queryFunction, errorHandler, successHandler) have up to four and three parameters respectively, only first one is mandatory. I rather use success and error handlers for tx.executeSql function, atomic error control could be better choice. Passing variables to functions: Only if another solution is not known or feasible, global variables are used: named after global_, and in block capitals. Deadline. Development was delayed because I have no enough spare time and above problems were time costly.

```
 ohcount -i  assets/www/js/database.js assets/www/js/interface.js \
 assets/www/js/create_populate_db.js  assets/www/index.html \
 assets/www/remove.html
Examining 5 file(s)
                              Ohloh Line Count
Language                Code    Comment  Comment %      Blank      Total  File
----------------  ---------  ---------  ---------  ---------  ---------  -------------
javascript             1374        155      10.1%        210       1739  database.js
javascript              393         91      18.8%         56        540  interface.js
javascript              402         57      12.4%         46        505  create_popula
html                    546         99      15.3%        117        762  index.html
javascript                1          0       0.0%          0          1  index.html
html                     28          1       3.4%          8         37  remove.html
```

Chapter 6

Conclusiones y trabajos futuros (5%)

Resumen de los logros principales conseguidos, destacando:

6.1 Implementación

...

6.2 Resultados

...

En futuros trabajos, a partir de una crítica constructiva del trabajo realizado, plantear mejoras y extensiones del mismo.

Bibliography

[Aca11] Academics. Classroom management system. https://play.google.com/store/apps/details?id=com.gradebook.academics, 2011.

[App05] Inc. Appcelerator. Aptana is a complete environment to build ror, php, python applications, along with complete html, css and javascript editing. http://www.aptana.org/, 2005.

[Aqu12] José Antonio Salgueiro Aquino. Classroom management for high school teachers. https://github.com/joseantoniosa/EduXes, 2012.

[Cas12] Manuel Rego Casasnovas. Application to access sergas appointment system. http://mrego.github.com/sergasapp/, 2012.

[dEXdG12] Consellería de Educación. Xunta de Galicia. Xade web is a web interface for educational data management. http://edu.xunta.es/xade/, 2012.

[fA10] Android for Academics. Attendance. http://androidforacademics.com/2010/09/attendance-instruction-manual/, 2010.

[Fou01] Eclipse Foundation. Eclipse interface development environment. http://www.eclipse.org/, 2001.

[Git08] GitHub. Powerful collaboration, review, and code management for open source and private development projects. https://github.com/, 2008.

[iagdc12] inpocketsolution at gmail dot com. Teacher aide. https://sites.google.com/site/teacheraidepro/, 2012.

[Inc01] Android Inc. Android simple development kit. http://developer.android.com/sdk/index.html, 2001.

[Lle12] Llerik. Teacher organizer. http://4pda.ru/forum/index.php?showtopic=345457&st=80, 2012.

[Mob12a] JQuery Mobile. Jquery mobile. http://jquerymobile.com/demos/1.1.1/, 2012.

[Mob12b] JQuery Mobile. The jquery mobile "page" structure. http://jquerymobile.com/demos/1.1.1/docs/pages/page-anatomy.html, 2012.

[Ora95] Oracle. Java standard edition. http://www.oracle.com/technetwork/java/javase/downloads/index.html, 1995.

[PÓ6] Ramón Castro Pérez. Classroom management web application. http://siestta.sourceforge.net/doc/index.html, 2006.

[Pho12a] PhoneGap. Phonegap. android getting started guide. http://docs.phonegap.com/en/1.8.1/guide_getting-started_android_index.md.html, 2012.

[Pho12b] PhoneGap. Phonegap getting started with android. http://docs.phonegap.com/en/1.8.1/guide_getting-started_android_index.md.html#Getting%20Started%20with%20Android, 2012.

[Pho12c] PhoneGap. Phonegap is a free and open source framework that allows you to create mobile apps using standardized web apis for the platforms you care about. http://www.phonegap.com, 2012.

[Pho12d] PhoneGap. Phonegap local storage. http://docs.phonegap.com/en/1.8.1/cordova_storage_storage.md.html#Storage, 2012.

[S.L09] Enxenio S.L. Complete classroom management system. http://www.sixa.es, 2009.

[Uni05] Unisoft. Académico ™is a software to school management. http://www.unisoftcolombia.com/unisoftcolombia/index.php?link=academico, 2005.