Detailed Design

Requested by: Dr. Scott Hunter
Professor of Computer Science
Computer Science Department of Siena College

Ms. Jami Cotler
Professor of Computer Science
Computer Science Department of Siena College

The Electronic Spreadsheet
Automated Teaching Assistant

Pear Software

Prepared by: Justin Valentini – Team Leader
Colin Cubinski
Daniel Nakhla
James Rocco
Matt Restivo

March 6, 2007
# e-SATA
## Detailed Design

### Table of Contents:

| Section 1.1: Product Overview and Summary | ................................................................. | 2 |
| Section 1.2: User Displays & Command Summary | ..................................................................... | 2 |
| Section 1.3: Detailed Data Flow Diagrams | ..................................................................... | 2 |
| Section 1.4: Logical Data Dictionaries | ..................................................................... | 2 |
| Section 1.5: Logical Data Stores | ..................................................................... | 2 |
| Section 1.6: Logical Format of Data Files and Databases | ......................................................... | 3 |

### Architectural Design Specification:

| Section 2.1: Structure Diagrams | ..................................................................... | 9 |
| Section 2.2: Use Case Scenarios | ..................................................................... | 14 |
| Section 2.3: Functional Requirements | ..................................................................... | 15 |

### Testing Requirements:

| Section 3.1: Acceptance Test | ..................................................................... | 19 |
| Section 3.2: Acceptance Test Example | ..................................................................... | 19 |
| Section 3.3: Unit Test | ..................................................................... | 21 |

### Detailed Design Specification

| Section 4.1: Documentation prologue for each routines | ..................................................................... | 22 |
| Section 4.2: Pseudocode for each routine | ..................................................................... | 22 |
| Section 4.3: Packaging specifications | ..................................................................... | 23 |

### Appendices:

| Section 5.1: Timeline | ..................................................................... | 24 |
| Section 5.2: Glossary of Terms | ..................................................................... | 25 |
Section 1.1: Product Overview and Summary

A computer automated spreadsheet file assessor and grader will take a major burden of busy college professors that do not have time to grade a large amount of assignments. Our clients, Dr. Scott Hunter and Ms. Jami Cotler have the problem of too many papers to grade and not enough time to do it. Pear Software can offer them a solution, one which will cut down on the tedium of grading Microsoft Excel Spreadsheet assignments for the Computer Science 010 class at Siena College, which a large amount of students are required to take. The e-SATA program will grade lab and pre-lab excel files using a master answer key.

The greater part of development will take place in the facilities provided for by Dr. Lederman, which is the Software Engineering Lab in Roger Bacon at Siena College. The operating of our software will take place in the Computer Science labs of Roger Bacon, in a Windows XP environment. Maintenance environments are not finalized at this stage, however they are assumed to be the similar machines as ones currently in the Roger Bacon Computer Labs.

Section 1.2: User Displays & User Command Summaries

See separate document: Pear Prototype Screens.

Section 1.3: Detailed Data Flow Diagrams

See separate document: Pear Data Flow Diagrams and Data Dictionaries.

Section 1.4: Data Dictionaries

See separate document: Pear Data Flow Diagrams and Data Dictionaries.

Section 1.5: Logical Data Stores

All information for e-SATA will be stored on an Oracle database. As per Dr. Hunter and Ms. Cotler, we will need to store submitted Excel files from all users, grading keys submitted from the Course Administrator, and grading specifications in a separate directory. As of now we plan to have the Oracle database contain only the locations the files rather than containing the files directly in the database.
Section 1.6: Logical Format of Data Files and Databases

The following is a list of tables, their elements, and the modules for which they will be used within e-SATA. Field corresponds to the variable name the elements are expected to have within that actual database.

Relationships:
### Attendance Table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AutoNum</td>
<td>Autonumber</td>
<td>Primary key. Will be filled in automatically by the database.</td>
</tr>
<tr>
<td>StudentID</td>
<td>Number</td>
<td>Foreign key. The ID of the user that attendance is being taken for.</td>
</tr>
<tr>
<td>Date</td>
<td>Text</td>
<td>Date that attendance is being taken.</td>
</tr>
<tr>
<td>Attendance</td>
<td>Text</td>
<td>Where the attendance is stored for the specific date. (e.g. Present or Not Present)</td>
</tr>
</tbody>
</table>

Modules: Lecture Instructor Attendance, Lab Instructor Attendance

- **field: AutoNum**
  - Primary key. Will be filled in automatically by the database.
  - type: int size: 10
- **field: StudentID**
  - Foreign key. Links each record of attendance to a student ID.
  - type: Integer
- **field: Date**
  - The date the attendance was taken.
  - type: varchar size: 10
  - *must be in mm/dd/yyyy format.
- **field: Attendance**
  - Where the attendance is stored for the specific date. (e.g. Present or Not Present)
  - type: varchar size: 10
  - *must be alphanumeric, being with character and be <= 10 varchars

### Courses Table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CourseID</td>
<td>Autonumber</td>
<td>Primary key. An autonumber given by the database.</td>
</tr>
<tr>
<td>CourseName</td>
<td>Text</td>
<td>Name of the course.</td>
</tr>
<tr>
<td>Semester</td>
<td>Text</td>
<td>Semester the course is offered.</td>
</tr>
<tr>
<td>Year</td>
<td>Number</td>
<td>Year the course is offered.</td>
</tr>
</tbody>
</table>

Modules: Course Administrator Create Class

- **field: CourseID**
  - Primary key. An autonumber given by the database.
  - type: Autonumber
- **field: CourseName**
  - A short description of the course.
  - type: varchar size: 50
  - *must be alphanumeric, being with character and be <= 50 varchars
- **field: Semester**
  - The semester the course takes place (e.g. Spring or Fall)
  - type: varchar size: 10
  - *must be alphanumeric, being with character and be <= 10 varchars
- **field: Year**
The year the course takes place (e.g. 2007)
type: Integer
*must be in yyyy format

Files Table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileID</td>
<td>AutoNumber</td>
<td>Primary key. A unique ID for the file.</td>
</tr>
<tr>
<td>FileName</td>
<td>Text</td>
<td>The name of the file.</td>
</tr>
<tr>
<td>AdminID</td>
<td>Number</td>
<td>Foreign key. The ID of the user who uploaded the file.</td>
</tr>
<tr>
<td>LabNumber</td>
<td>Number</td>
<td>The lab number.</td>
</tr>
<tr>
<td>CourseID</td>
<td>Number</td>
<td>Foreign key. ID of the course to which the file is linked.</td>
</tr>
<tr>
<td>Type</td>
<td>Text</td>
<td>Template or Answer Key</td>
</tr>
</tbody>
</table>

Modules: Course Administrator Upload Template File / Answer Key

field: FileID
Primary key. Will be filled in automatically by the database.
type: AutoNumber

field: FileName
The name of the file that was uploaded to the system.
type: varchar size: 50
*must be alphanumeric, being with character and be <= 50 varchars

field: AdminID
Foreign key. Links each uploaded file to the ID of the person that uploaded the file.
type: Integer

field: LabNumber
The number of the lab to which the file corresponds.
type: Integer

field: CourseID
Foreign key. Links each uploaded file to a specific course ID.
type: Integer

field: Type
The type of file that was uploaded. (e.g. Template or Answer Key)
type: varchar size: 15
* must be alphanumeric, being with character and be <= 15 varchars
**Instructs Table:**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SectionID</td>
<td>Number</td>
<td>Primary key. The ID of the section which a professor is teaching.</td>
</tr>
<tr>
<td>ProfessorID</td>
<td>Number</td>
<td>Foreign key. The ID of the user who is teaching the section.</td>
</tr>
</tbody>
</table>

Modules: Lecture Instructor, Lab Instructor, and Course Administrator Account Associations.

- **field:** SectionID
  - Primary key and foreign key. Links to a SectionID from the Sections table.
  - type: Integer

- **field:** ProfessorID
  - Foreign key. Links each professor’s ID to one or more sections.
  - type: Integer

**Labs Table:**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LabID</td>
<td>AutoNumber</td>
<td>Primary key. Will be filled in automatically by the database.</td>
</tr>
<tr>
<td>FileName</td>
<td>Text</td>
<td>Name of the file.</td>
</tr>
<tr>
<td>StudentID</td>
<td>Number</td>
<td>Foreign key. ID of the student who uploaded the file.</td>
</tr>
<tr>
<td>EditedBy</td>
<td>Number</td>
<td>Foreign key. ID of the instructor that edited the grade.</td>
</tr>
<tr>
<td>SystemGrade</td>
<td>Text</td>
<td>Lab or Pre-Lab</td>
</tr>
<tr>
<td>Type</td>
<td>Text</td>
<td>The grade given by the e-SATA application.</td>
</tr>
<tr>
<td>PartialCredit</td>
<td>Number</td>
<td>Any partial credit given for this file.</td>
</tr>
<tr>
<td>GradeWeight</td>
<td>Number</td>
<td>How much of the total points this file is worth. (0.0 being 0% and 1.0 being 100%)</td>
</tr>
<tr>
<td>Feedback</td>
<td>Text</td>
<td>Any feedback or comments made by an instructor.</td>
</tr>
</tbody>
</table>

Modules: Student, Lecture Instructor, Lab Instructor, and Course Administrator.

- **field:** LabID
  - Primary key. Will be filled in automatically by the database.
  - type: AutoNumber size: 10

- **field:** FileName
  - The name of the lab file that was uploaded to the system.
  - type: varchar size: 50
  - *must be alphanumeric, being with character and be <= 50 varchars

- **field:** StudentID
  - Foreign key. Links each uploaded lab to the student that uploaded the lab.
  - type: Integer

- **field:** EditedBy
  - Foreign key. If a grade has been changed then this field will contain the ID of the lab instructor that changed the grade.
  - type: Integer

- **field:** Type
  - The type of file that was uploaded. (e.g. Pre-Lab or Lab)
  - type: varchar size: 10
field: SystemGrade
   The grade assigned to the lab by the e-SATA system.
type: Decimal
   *must be positive
field: PartialCredit
   Any partial or extra credit that was given for this lab to the student.
type: Decimal
field: GradeWeight
   The amount this file is worth when compared to the lab as a whole.
type: Decimal
   *value must be between 0.0 and 1.0
field: Feedback
   A description of any mistakes or comments on the lab.
type: varchar size: 250
   *must be alphanumeric, being with character and be <= 250 varchars

Sections Table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SectionID</td>
<td>Number</td>
<td>Primary key. A unique ID for the section.</td>
</tr>
<tr>
<td>CourseID</td>
<td>Number</td>
<td>Foreign key. The ID of the course to which the section is linked.</td>
</tr>
<tr>
<td>Type</td>
<td>Text</td>
<td>Lab or Lecture.</td>
</tr>
</tbody>
</table>

Modules: Lecture Instructor, Lab Instructor, and Course Administrator Account Associations.

field: SectionID
   Primary key and foreign key. The section number.
type: Integer
field: CourseID
   Foreign key. Links the section to a specific course.
type: Integer
field: Type
   Contains the type of section (e.g. Lab or Lecture)
type: varchar size: 20
   *must be alphanumeric, being with character and be <= 20 varchars

Users Table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserID</td>
<td>AutoNumber</td>
<td>Primary key. A unique ID number given to every registered user. Primary key.</td>
</tr>
<tr>
<td>UserName</td>
<td>Text</td>
<td>The UserName is the person's email address.</td>
</tr>
<tr>
<td>Password</td>
<td>Text</td>
<td>Encrypted password.</td>
</tr>
<tr>
<td>FirstName</td>
<td>Text</td>
<td>The user's first name.</td>
</tr>
<tr>
<td>LastName</td>
<td>Text</td>
<td>The user's last name.</td>
</tr>
<tr>
<td>Type</td>
<td>Number</td>
<td>Designates the users type (Student, Lecture Instructor, Lab Instructor, Course Admin)</td>
</tr>
</tbody>
</table>
Modules:

- Web Interface User Login, Grading Application User Login.

field: UserID
- Primary key.
- type: AutoNumber size: 20

field: UserName
- The user’s email address.
- type: varchar size: 50
- *must be alphanumerics, being with character and be <= 50 varchars

field: Password
- The student’s password.
- type: varchar size: 255 *size large for encryption
- *must be alphanumerics and be <= 20 varchars

field: FirstName
- The first name of the student.
- type: varchar size: 40
- *must be only characters, <= 40 varchars

field: LastName
- The last name of the student.
- type: varchar size: 50
- *must be only characters, <= 50 varchars

field: Type
- Contains the type of user (Student, Lab, Lecture, or Course Admin).
- type: Integer
Section 2.1: Structure Diagrams

Structure charts are graphical representations similar to that of the Data Flow Diagrams. They depict the subroutine and function hierarchy in the program. By viewing structure charts, one can identify and fully understand the manner in which the program is set up.

Key

MODULE

LIBRARY MODULE
Lecture Instructor

- Create Account
- Change Account Information
- View Student Requests
- Login
  - View Grades of Students in a course
- Retrieve Forgotten Password
  - Drop Student from Section
  - Compare Student Mistakes
Section 2.2: Use Case Scenarios

There will four determined users of the e-SATA application. This is determined in a fashion so that proper user rights and functions are assigned as there will be more than one type of client using the application. The following four users will be:

- Student
- Lab Instructor
- Lecture Instructor
- Course Administrator

Student

Students will be able to make submissions of their pre-lab and labs files. After submitting their pre-lab they will also have the right to resubmit the pre-lab if they want to make changes before the lab session after feedback is received. The feedback screen will provide comments to incorrect cells to show mistakes; if a mistake is not limited to one cell there will be one dedicated cell to hold all of the notes on these mistakes. If a mistake is limited to only one cell a comment will be added to incorrect cells directly. Students will also have the ability to view their own lab grades. If a student misses a lab and attends another to make it up, the software will be able to recognize the discrepancy and resume normal function with a lab instructor override.

Lab Instructor

Lab Instructors will be able to view the profile of any student in a lab session. A profile encompasses the grades and attendance of a student. Viewing the profile signifies the ability to view and edit the students’ grade. e-Sata will grade the pre-lab and lab, and then the lab instructors will scan over the e-Sata assessment and adjust anything that was marked incorrectly (i.e. formulas), or assigning partial credit as necessary. Lab Instructors will also be able to get feedback on what errors or mistakes made by students have the highest percentage. Feedback from other lab sessions will also be available to Lab Instructors. Lab Instructors will be able to keep attendance using the application and perform overrides.

Lecture Instructor

Lecture Instructors will have the right to view their students’ grades and view statistics and feedback from all lab sessions. The feedback includes what errors or mistakes made by students have the highest percentage.

Course Administrator

The Course Administrator will be able to view all the grades from all the sections. The Course Administrator will also have full access rights to edit and upload answer keys used by e-Sata to grade the pre-labs and labs.
Section 2.3: Functional Requirements

This section is an outline of the Use Case scenarios seen above. It describes the functions of the four types of users and possible options and actions that they have while using the system. They are outlined in the following manner: Student, Lab Instructor, Lecture Instructor, Course Administrator.

I) Student:

- Create an account with the following information:
  - First Name.
  - Last Name.
  - Email.
    - The system will use the email address as the user name which will be required when logging into the system.
    - The system will verify the email address is not already in use or taken by another account in the system.
  - Password.
  - Lab Section.
  - Lecture Section.
  - A confirmation email will be sent containing account information.
- Login in using email and password.
- Retrieve a forgotten password.
  - The password will be emailed to the user’s email account.
- Change the following account information:
  - Password.
    - The new password will be emailed to the user’s account.
  - First Name.
  - Last Name.
  - Lab Section.
  - Lecture Section.
- Download template files from the system for pre-labs and labs.
- View attendance.
- Upload a pre-lab or lab to the system.
  - Select the pre-lab or lab number to upload.
  - Browse to the pre-lab or lab file.
  - The system will verify the file is of the appropriate extension.
  - The system will verify the file name conforms to a standard naming convention.
- View feedback of a submitted pre-lab.
- Resubmit a graded pre-lab.
- View feedback on submitted pre-labs and labs.
  - View grades of submitted labs.
  - Select a specific pre-lab or lab to view the following feedback:
    - The submitted file.
    - List of errors and mistakes.
      - Location of error.
• Type of error.
  o Download from the system any submitted pre-labs or labs.
• View any labs that the student may have missed.
  o View if the lab has been made up and what teacher and lab section the lab has been made up with.

II) Lab Instructor:

• Login in using an email and password.
  o Required to change password on first login.
• Retrieve a forgotten password.
  o The password will be emailed to the user’s email account.
• Change the following account information:
  o Password.
    • The new password will be emailed to the user’s account.
• View a list of all students they teach with the ability to sort by name, grade, and section.
  o Ability to drop a student from their lab sections.
  o Accept or deny pending changes of sections or labs made by students.
  o Ability to filter students by section.
  o Ability to sort by name and/or grade.
• Select an individual student to view the profile of the student.
  o View attendance of that student.
  o View grades for that student.
  o Select whether or not a student has submitted a pre-lab.
  o Change grades for a particular lab.
    • Add in extra points for things such as Skills Checks.
    • Add in partial credit.
  o View markup of each lab that the student has completed.
    • Markup consists of mistakes or missed points.
  o Ability to drop the student from their section.
• View student requests to change lab section to a section taught by the lab instructor.
  o View all pending requests.
  o Ability to approve or deny request to change lab section.
• View a list of the most common mistakes for a specific lab from their lab sections.
  o Ability to filter on a particular section.
• Keep attendance for every student in a lab instructor’s sections.
  o The system will allow the lab instructor to input one of three choices for each student’s attendance (present, not present, late).
  o The system will show an absence for a specific date when the students logs in.
III) Lecture Instructor:

- Login in using a user name and password.
  - Required to change password on first login.
- Retrieve a forgotten password.
  - The password will be emailed to the user’s email account.
- Change the following account information:
  - Password.
    - The new password will be emailed to the user’s account.
- View a list of all students they teach with the ability to sort by name, grade, and section.
  - Ability to drop a student from their lab sections.
  - Accept or deny pending changes of sections or labs made by students.
  - Ability to filter students by section.
  - Ability to sort by name and/or grade.
- Select an individual student to view the profile of the student.
  - View attendance of that student.
  - View grades for that student.
  - View markup of each lab that the student has completed.
    - Markup consists of mistakes or missed points.
  - Ability to drop the student from their section.
- View student requests to change lecture section to a section taught by the lecture instructor.
  - View all pending requests.
  - Ability to approve or deny request to change lecture section.
- View a list of the most common mistakes for all students for a specific lab from any of the user’s lecture sections.
  - Ability to filter on a particular section.
- Select a specific student to view their mistakes compared to common mistakes of other students in that student’s section.

IV) Course Administrator:

- Login in using a user name and password.
- Create the following new user accounts:
  - Student.
  - Lab Instructor.
  - Lecture Instructor.
  - Course Administrator.
- View a table of Students and their information.
  - Filter and/or sort results based on:
    - Course.
    - Lecture section.
    - Lab section.
    - Pre-lab or lab number.
      - Grade for the pre-lab or lab.
• Grade average for all pre-labs or labs.

• View a table of Lab Instructors and their information.
  o Filter and/or sort results based on:
    ▪ Courses taught.
    ▪ Lab sections taught.

• View a table of Lecture Instructors and their information.
  o Filter and/or sort results based on:
    ▪ Courses taught.
    ▪ Lecture sections taught.

• View a table of Course Administrators and their information.

• Select an individual profile for detailed account information.
  o View the user name and password.
  o Change the user’s password.
  o Change the user’s account type. (e.g. Change an account from Lab Instructor to Lecture Instructor.)
  o Change any other account information such as email address.

• Manage a user’s account.
  o Edit any information corresponding to the account.
  o Edit any lab or lecture sections they are associated with.
  o Delete the account.

• Create a new course with a unique course number.
  o Assign a year and semester to a specific course.

• Create a new lecture section.
  o Assign the lecture section to a course.
  o Assign a Lecture Instructor to the lecture section.
  o Assign a schedule for the section which consists of:
    ▪ The day(s) that the lecture meets. (e.g. MWF)
    ▪ The time that the lecture meets. (e.g. 8:30 AM – 9:25 AM)
    ▪ The system will check that the meeting day(s) and time do not conflict with another lab or lecture section.
  o Assign a room number where the lecture is held.

• Manage lecture sections.
  o Edit a lecture section which includes the following information:
    ▪ The course it is assigned to.
    ▪ The day(s) the lecture is held.
    ▪ The time the lecture is held.
    ▪ The room where the lecture is being held.
  o Delete a linked lecture section.

• Create a new lab section.
  o Assign the lab section to a course.
  o Assign a Lab Instructor to the lab section.
  o Assign a schedule for the section which consists of:
    ▪ The day(s) that the lab meets. (e.g. MWF)
    ▪ The time that the lab meets. (e.g. 8:30 AM – 9:25 AM)
    ▪ The system will check that the meeting day(s) and time do not conflict with another lab or lecture section.
- Assign a room number where the lab is held.
- Manage lab sections.
  - Edit a lab section which includes the following information:
    - The course it is assigned to.
    - The day(s) the lab is held.
    - The time the lab is held.
    - The room where the lab is being held.
  - Delete a linked lab section.
- Manage answer keys for e-SATA which are used for the automated grading of pre-labs and labs.
  - Upload answer keys.
    - The system will check that the answer key is a text file.
    - The system will check that the file name conforms to a standard naming convention.
      (e.g. Lab01Key.txt)
    - The system will check that the answer key contains the following information in a uniform format:
      - A code to distinguish the type of problem that needs to be graded. (e.g. F for formatting, S for skills check)
      - The cell to look at when grading the type of problem noted above. (e.g. B6)
      - The grade weight of the problem
      - Any other pertinent information.
  - Delete any previously uploaded answer keys.
  - Assign a specific answer key to a lab or pre-lab in a course.
- Manage template files.
  - Assign the template file to a course number.
  - Assign the template file to a pre-lab or lab number.
  - Delete a template file.

Use Case Narrative:

e-SATA will be a web-based application that will allow the user to login and perform specific tasks as per their designation as either a student, lab instructor, lecture instructor, or course administrator. e-SATA will communicate directly with a database, being able to submit and retrieve information at the request of the user, assuming the user has the privileges to access that material. Functions include submission of assignments for automated grading, the ability to view or edit grades and attendance, and to view feedback on graded assignments.

Section 3.1: Acceptance Test

See separate document: Pear Test Plan.

Section 3.2: Acceptance Test Example
<table>
<thead>
<tr>
<th>Item Being Tested</th>
<th>Testing For</th>
<th>Expected Outcome</th>
<th>Pass</th>
<th>Fail</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 Login Screen</td>
<td>Does the screen load?</td>
<td>The screen should load correctly and users will have the ability to type in their login information.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0 New User Screen</td>
<td>Does the screen load?</td>
<td>The screen should load correctly and users will have the ability to type in their personal information.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.0 Password Retrieval Screen</td>
<td>Does the screen load?</td>
<td>The screen should load correctly and users will have the ability to type in their username to retrieve the lost password.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.0 Student User Screen</td>
<td>Does the screen load?</td>
<td>The screen should load correctly and students will have access to all functions designated to their user type.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.1 Student User Screen</td>
<td>Are all features available?</td>
<td>All tabs, buttons, and fields should be accessible and clickable.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.2 Student User Screen</td>
<td>Lab Grade Screen</td>
<td>All lab grades for the user should be displayed.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.3 Student User Screen</td>
<td>Pre-Lab Submission Screen</td>
<td>The user can view pre-lab grades and has the option to re-submit files.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.4 Student User Screen</td>
<td>Submit File Screen</td>
<td>The user has the option to upload any file for grading.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.5 Student User Screen</td>
<td>Feedback Screen</td>
<td>The user has the option to download feedback from any assignment.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.0 Lab Instructor Screen</td>
<td>Does the screen load?</td>
<td>All tabs, buttons, and fields should be accessible and clickable.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.1 Lab Instructor Screen</td>
<td>Grading Screen</td>
<td>The user can choose a specific lab session that allows them to view/edit the student’s grades and attendance.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.2 Lab Instructor Screen</td>
<td>Feedback Screen</td>
<td>The screen displays the option to select a lab, and view individual student or class errors.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.0 Lecture Instructor Screen</td>
<td>Does the screen load?</td>
<td>All tabs, buttons, and fields should be accessible and clickable.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Section 3.3: Unit Test

See separate document, Pear Test Plan for a more in depth description of the unit test.

The Unit test is a complete breakdown of all functionalities on every GUI page. It tests every option for features such as buttons, text boxes, and any other way that a user may interact with the interface. If one test fails, the entire section fails.

<table>
<thead>
<tr>
<th>6.1 Lecture Instructor</th>
<th>Grade Screen</th>
<th>The user gets a screen displaying student’s grades.</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2 Lecture Instructor</td>
<td>Feedback Screen</td>
<td>The screen displays the option to select a lab, and view individual student or class errors.</td>
</tr>
<tr>
<td>7.0 Course Administrator</td>
<td>Does the screen load?</td>
<td>All tabs, buttons, and fields should be accessible and clickable.</td>
</tr>
<tr>
<td>7.1 Course Administrator</td>
<td>Grade Screen</td>
<td>The user has the option to choose a specific lab session that allows them to view/edit the student’s grades.</td>
</tr>
<tr>
<td>7.2 Course Administrator</td>
<td>Feedback Screen</td>
<td>The screen displays the option to select a lab, and view individual student or class errors.</td>
</tr>
<tr>
<td>7.3 Course Administrator</td>
<td>Answer Key Screen</td>
<td>This screen gives the user an option to upload pre-lab/lab answer keys.</td>
</tr>
</tbody>
</table>
Section 4.1: Documentation prologue for each routines

Function 1) validLogin

Input: User’s Information
Output: Forward user to Default Screen

This function takes all of the information entered by the user on e-SATA’s default screen, the login screen, and compares it to the entries stored in the systems database. If the information matches, it is considered a valid login, and the user loads to their default profile screen. Otherwise, the login is invalid and they must re-enter their info.

Function 2) uploadFile

Input: File Path
Output: Upload Confirmation

This function takes the path navigated to by the user, and uploads the file at the location to the e-SATA database.

Function 3) cellFont

Input: Cell
Output: String

This function takes the information stored at a specific cell in the spreadsheet, and compares it to the correct answer. If the font in the cell matches that on the answer key, the student is awarded points, and if it doesn’t, they are deducted.

Function 4) sortName

Input: User Click(Column Header)
Output: Sorted List

This function takes a specific list of data on the screen, and sorts it in ascending or descending order, according to the number of clicks.

Section 4.2: Pseudocode for each routine

Function 1) Psuedocode:

Compare user name, password
Open connection to database
If user name, password == user name, Password in the database
Allow user access
Else
Reject access

**Function 2) Psuedocode:**
Store file path
Extract file from source, store duplicated in database

**Function 3) Psuedocode:**
Store data in current cell
Load correct answer, store data
Compare both data variables
If equal,
Increase point counter
Else
Decrease point counter

**Function 4) Psuedocode:**
The user clicks on a column header
Define counter
If counter mod 2 is 0
Sort list ascending
Else if counter mod 2 is 1
Sort list descending
Else
Default list display

---

**Section 4.3: Packaging specifications**

Pear Software will install its XLS Automatic Excel Grading System and all necessary files onto the Siena College Computer Science Department’s web server, Oraserv. For easy movement of the application in the future, all of the path extensions will be relative, rather than absolute. There will then be a web address supplied for users to access our program.
Section 5.1: Timeline
Section 5.2: Glossary of Terms

**Code** - The symbolic arrangement of data or instructions in a computer program or the set of such instructions.

**Database** - A collection of data arranged for ease and speed of search and retrieval.

**DBMS (Database Management System)** - Software that controls the organization, storage, retrieval, security and integrity of data in a database.

**Gantt Chart** - A chart that depicts progress in relation to time, often used in planning and tracking a project.

**HTML (Hyper Text Markup Language)** – A markup language used to structure text and multimedia documents and to set up hypertext links between documents, used extensively on the World Wide Web.

**Internet** - An interconnected system of networks that connects computers around the world via the TCP/IP protocol.

**Linear Sequential Model / Classic Waterfall Model** – A systematic, sequential approach to software development that begins at the system level and progresses through analysis, design, coding, testing, and support.

**PC (Personal Computer)** - Another name for a microcomputer designed for use by a single user.

**PHP (PHP Hypertext Preprocessor)** – A server-side, cross-platform, HTML-embedded scripting language used to create dynamic web pages. PHP is open source software.

**Software** - Written programs or procedures or rules and associated documentation pertaining to the operation of a computer system and that are stored in read/write memory.

**SQL (Structured Query Language)** - Pronounced "sequel", it is a language that provides an interface to relational database systems.

**TCP/IP (Transmission Control Protocol / Internet Protocol)** - the suite of communications protocols used to connect hosts on the Internet. TCP/IP uses several protocols, the two main ones being Transmission Control Protocol (TCP) and Internet Protocol (IP).