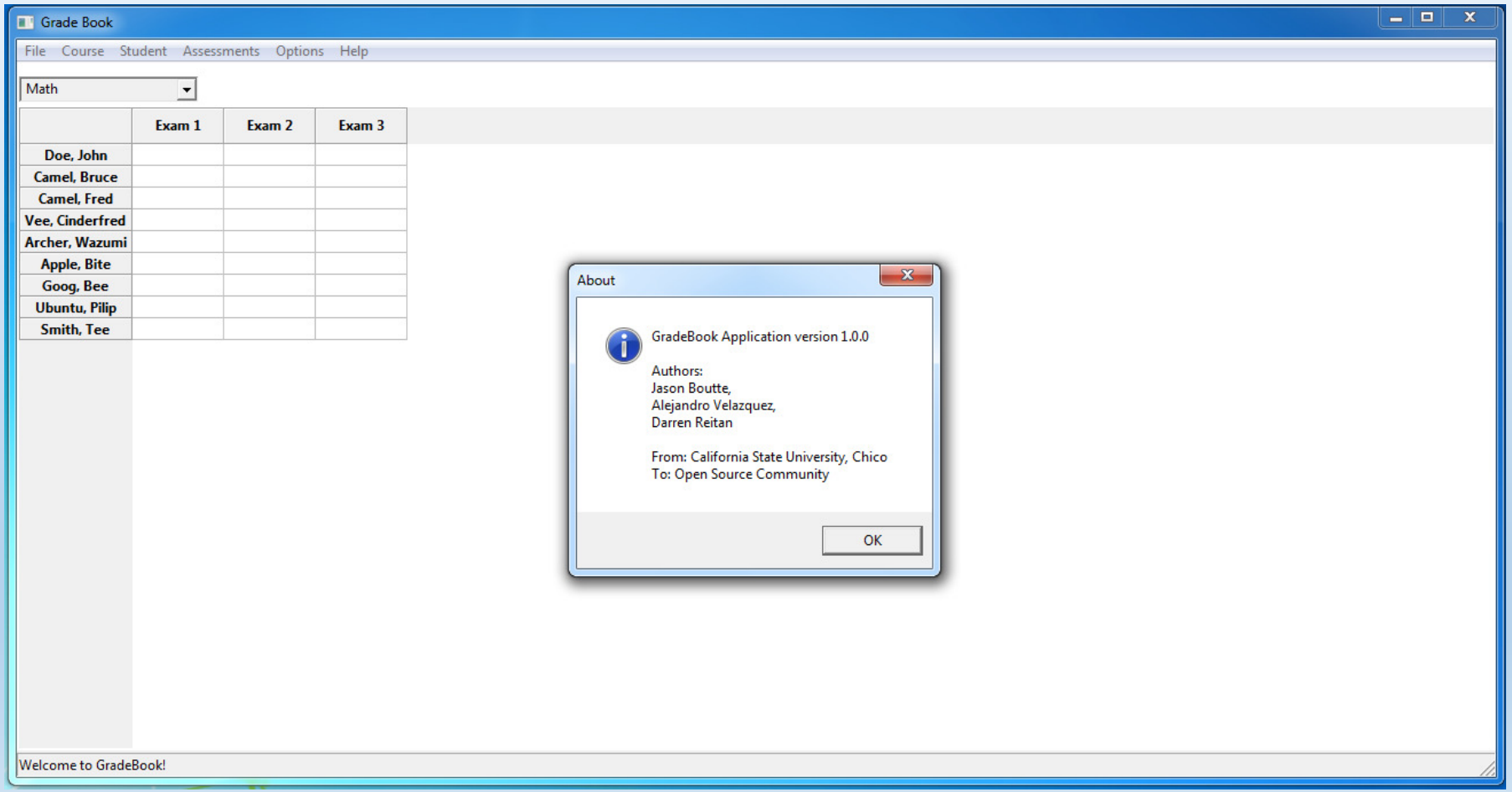
Team Members:

- **Jason Boutte**
- **Alejandro Velazquez**
- **Darren Reitan**

Welcome to Gradebook!



Overview

Gradebook targets …

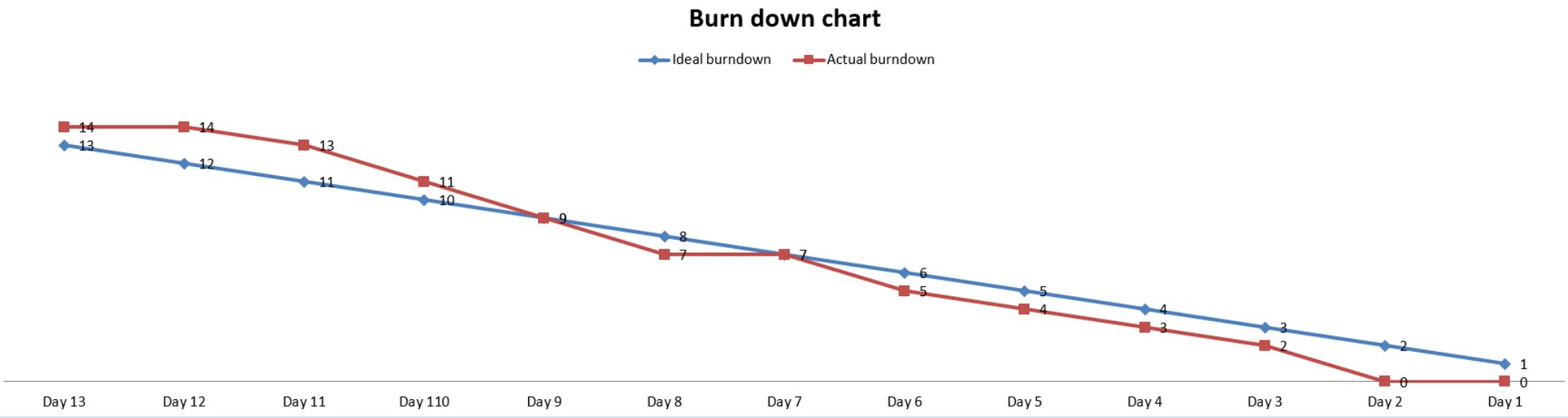
- *Instructors at any educational institution.*
- *Beginners to advance computer users.*

Gradebook can …

- Keeps track of students, their grades, and assessments.
- Easily create graphs.
- Imports students from either *Blackboard Learn* or *Moodle.*
- Runs on a Windows or Unix based operating system.

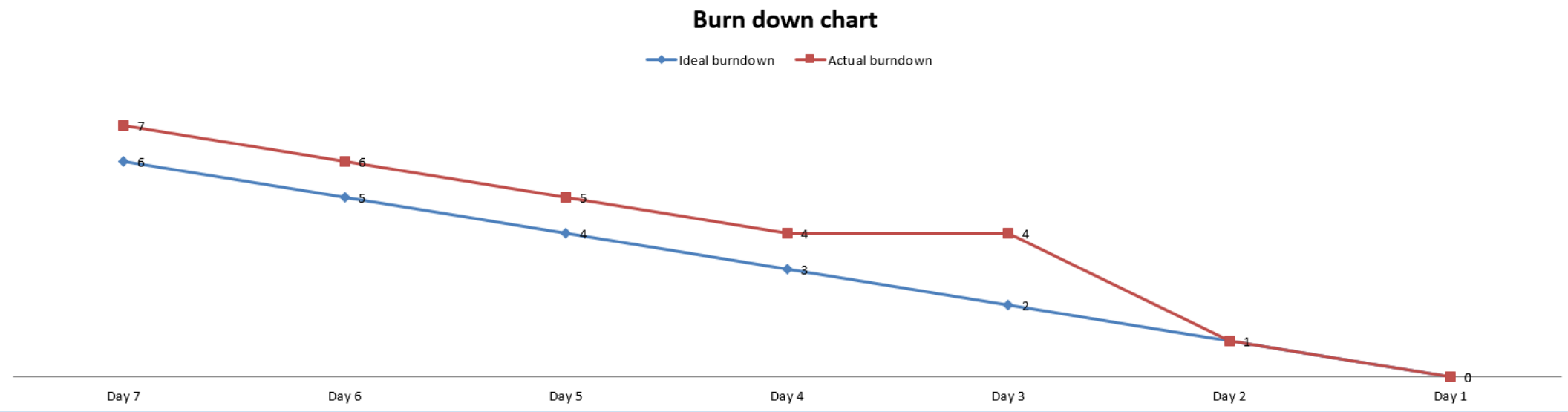
Sprint: September 8th – 24th
"As a Teacher, I want to add multiple courses."

1. Create File Menu GUI to add courses.
2. Create SQL interface for storing courses.
3. Connect SQL with GUI.



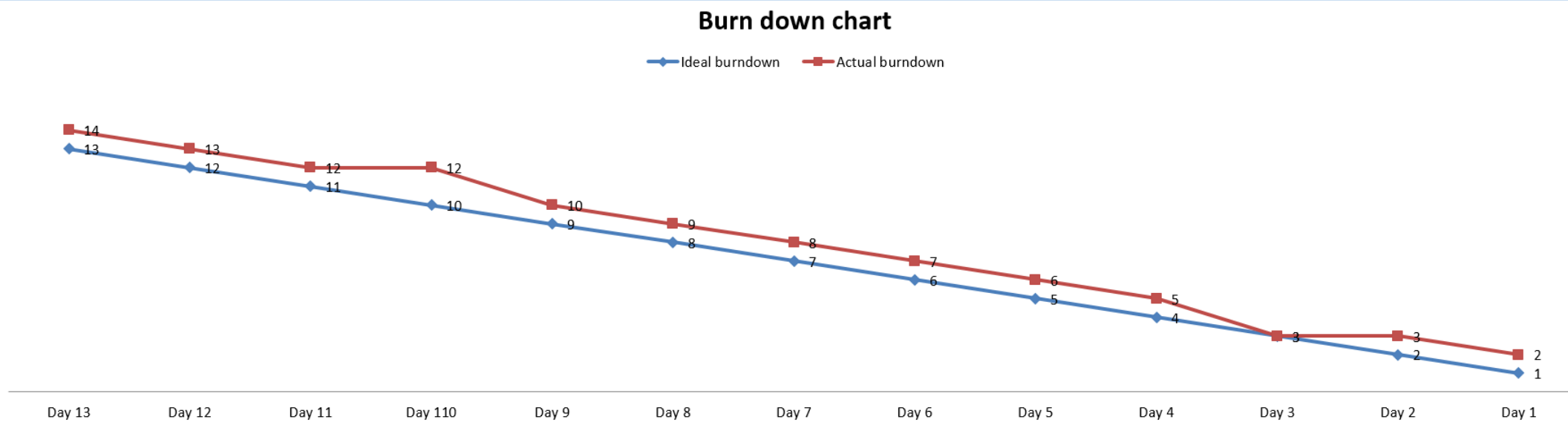
Sprint: September 25th – October 12th
"As a Teacher, I want to add students to my course."

1. Create GUI to add Students to Courses.
2. Add SQL interface for inserting Students into a Course.
3. Connect GUI and SQL code.
4. Add Students to Course.



Sprint: October 13th – October 27th
"As a Teacher, I want to add assessments."

1. Create GUI to add Assessments.
2. Add SQL code for Assessments.
3. Connect GUI and SQL code for Assessments.
4. Add Assessments to class.



Design

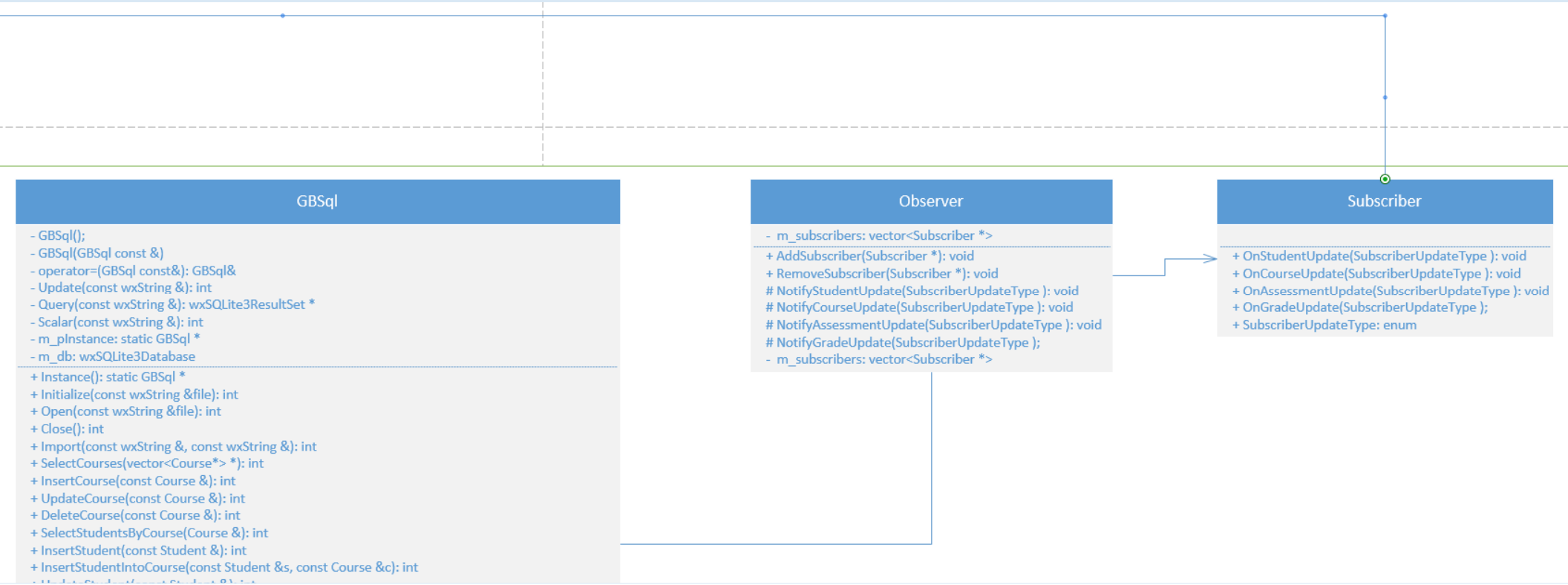
- **Singleton Pattern**
- **Observer Pattern**
- **Model-View-Controller (MVC) Architecture**

Sprint: October 28th – November 10th
Implementing: Singleton, Observer, and MVC

- **Singleton Pattern**
  - **Why?**

Sprint: October 28th – November 10th
Implementing: Singleton, Observer, and MVC

- **Observer Pattern**



Sprint: October 28th – November 10th
Implementing: Singleton, Observer, and MVC

- **Observer Pattern cont.**

```
12  /**
13    * @brief  Constructor to connect GBFrameController with a view.
14    * @param  GBFrameView * view: The view to connect the Controller to.
15    * @retval none.
16    */
17  GBFrameController::GBFrameController(GBFrameView *view)
18    : m_pMainFrameView(view),
19      m_pSql(GBSql::Instance()) {
20
21    m_pSql->AddSubscriber(this);
22
23    PopulateCourseDropDownList();
24    (m_pMainFrameView->m_pCourseComboBox)->SetSelection(0);
25    CreateGridView();
26  }
```

Sprint: October 28th – November 10th
Implementing: Singleton, Observer, and MVC

- **Model-View-Controller (MVC)**

View

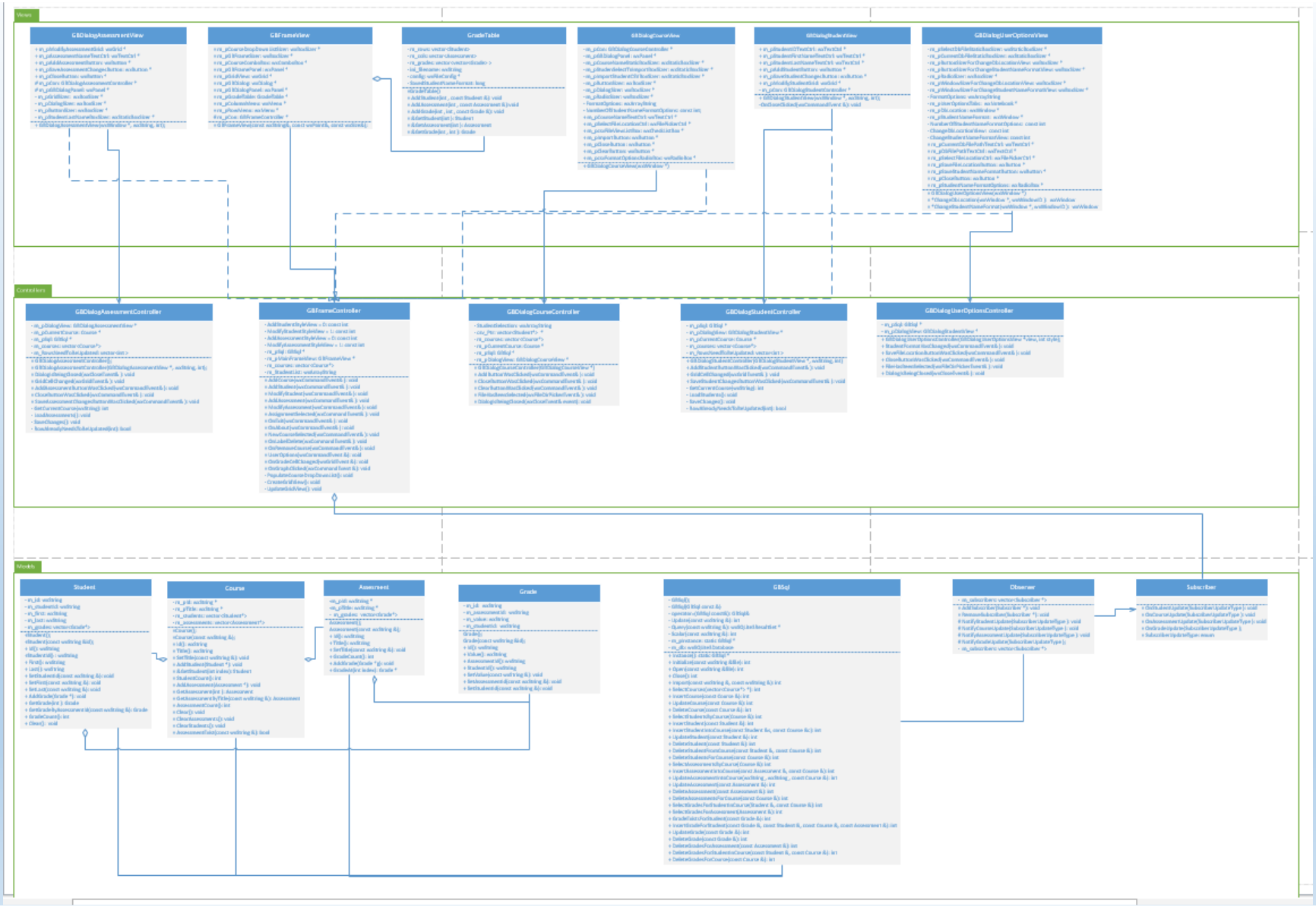
- GBDialogAssessmentView,
- GBDialogCourseView,
- GBDialogUserOptionsView
- GBFrameView,
- GBDialogStudentView,

Controller

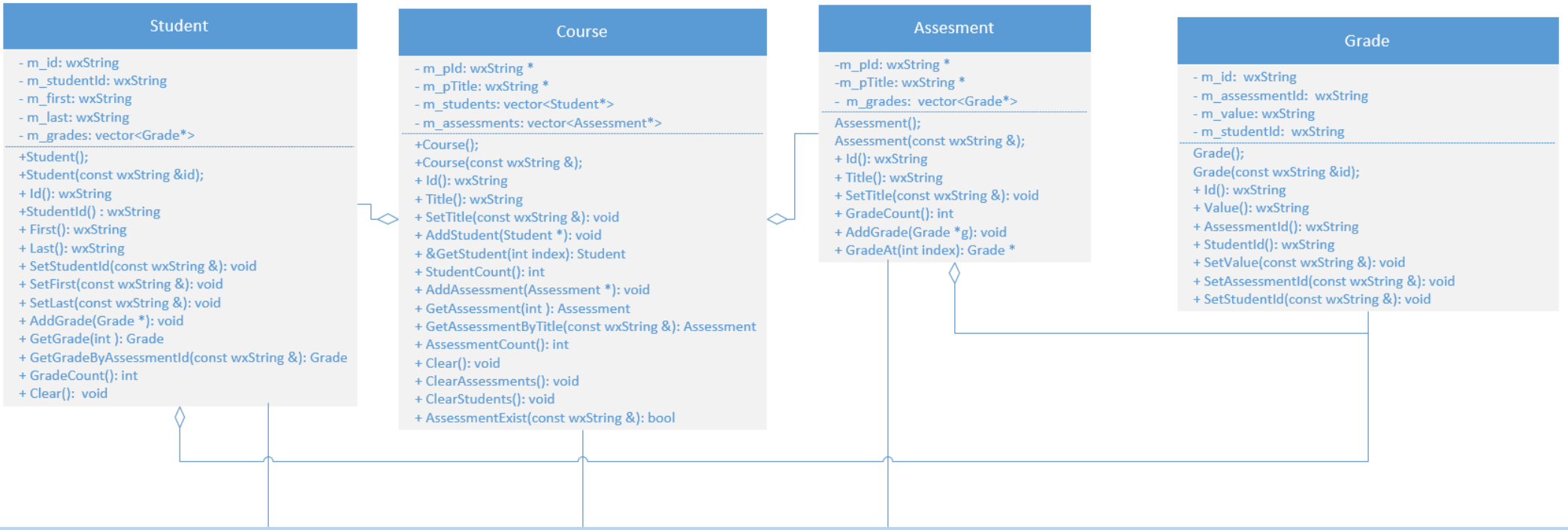
- GBDialogAssessmentController,
- GBDialogCourseController,
- GBDialogUserOptionsController
- GBFrameController,
- GBDialogStudentController,

Model

- Subscriber,
- Observer,
- GBSql,
- Course,
- Grade,
- Assessment
- Student
- bbimporter


Views
GBDialogAssessmentView
GBFrameView
GradeTable
GBDialogCourseView
GBDialogStudentView
GBDialogUserOptionsView
Controllers
GBDialogAssessmentController
GBFrameController
GBDialogCourseController
GBDialogStudentController
GBDialogUserOptionsController
Models
Student
Course
Assessment
Grade
GBSql
Observer
Subscriber

Model Relationships



Database Schema

courses		
id	integer	primary key
title	text	not null
students		
id	integer	primary key
sid	text	not null
first	text	not null
last	text	not null
assessments		
id	integer	primary key
title	text	not null
cid	text	not null
grades		
id	integer	primary key
sid	text	not null
cid	text	not null
aid	text	not null
value	text	
adj_value	text	
course_student		
id	integer	primary key
sid	text	not null
cid	text	not null

Testing and Verification

We have Unit Tests that will …

- Verify that **inserting, deleting, and updating information is always working** accordingly.
- Verify that **no relationships in the Database are broken.**

```
168  TEST_F(GBSqlTest, InsertStudentIntoCourse) {
169      Student s;
170
171      s.SetStudentId(kStudentId);
172      s.SetFirst(kFirst);
173      s.SetLast(kLast);
174
175      Course c("1");
176
177      int r = sql->InsertStudentIntoCourse(s, c);
178
179      EXPECT_EQ(1, r);
180
181      wxSQLite3ResultSet *rs = Query("SELECT * FROM students");
182
183      rs->NextRow();
184
185      ASSERT_STREQ(kStudentId.mb_str(), rs->GetString("sid").mb_str());
186      ASSERT_STREQ(kFirst.mb_str(), rs->GetString("first").mb_str());
187      ASSERT_STREQ(kLast.mb_str(), rs->GetString("last").mb_str());
188
189      rs = Query("SELECT * FROM course_student");
190
191      rs->NextRow();
192
193      ASSERT_STREQ(kStudentId.mb_str(), rs->GetString("sid").mb_str());
194      ASSERT_STREQ("1", rs->GetString("cid").mb_str());
195  }
```

```
418  TEST_F(GBSqlTest, DeleteGradesForStudentInCourse) {
419      DefaultGrade();
420
421      Student s("1");
422
423      s.SetStudentId(kStudentId);
424
425      Course c("1");
426
427      int r = sql->DeleteGradesForStudentInCourse(s, c);
428
429      EXPECT_EQ(1, r);
430      ASSERT_EQ(0, GradeCount());
431  }
```

Metric++ checks for …

- Architecture quality
- Design quality
- Code quality
- Test quality

Cyclomatic Complexity Average = 0.830039526

Demo