**Teach Easy**

**Bits Please!**

**Technical Manual**

**Last Update: 12/02/2018**

**Table of Contents**

**1.1.0 Glossary**

**2.1.0 Contributors**

**3.1.0 File Descriptions**

**4.1.0 Database**

**4.1.1 Definition**

**4.1.2 Tables**

**4.1.3 Instructions/Implementing**

**5.1.0 Calendar**

**5.1.1 License**

**5.1.2 Bootstrap**

**5.1.3 JavaScript**

**5.1.4 Events**

**Adding Events**

**Clicking an Event**

**5.1.5 Lesson Plan**

**6.1.0 Attendance**

**6.1.1 Dependencies**

**6.1.2 Data Tables**

**Attendance Tab Splash Page**

**6.1.3 Queries**

**Attendance Tab-Main Page**

**Attendance Tab-Take Attendance**

**Attendance Tab-Take Attendance, Insert Absences**

**Attendance Tab-Take Attendance, Insert Absences Check if it already exists**

**Attendance Tab-Remove Single Absence Modal**

**Attendance Tab-Add Single Absence Modal**

**6.1.4 Modals**

**6.1.5 Take Attendance Page**

**7.1.0 Gradebook**

**7.1.1 Table**

**7.1.2 Grade Entry**

**7.1.3 Queries**

**Displaying Grades**

**Adding New Assignments**

**8.1.0 Website**

**1.1.0 Glossary**

***Query*** - Commands used to retrieve information from a database.

***Modal*** - A scripted effect that allows you to overlay a small element over a website.

***Database*** - A collection of data that is specially organized for rapid search and retrieval by a computer.

***ER Model*** - Entity-relationship model, a graphical representation of entities and their relationships to each other, typically in regard to organization of data within databases.

***Primary Key*** - A special relational database table column designated to uniquely identify all table records.

***Foreign Key*** - A field in one table that uniquely identifies a row of another table or the same table in a database.

***One to One Relationship*** - A type of cardinality that refers to the relationship between two entities A and B in which one element of A may only be linked to one element of B, and vice versa.

***One to Many Relationship*** - A parent record in one table can potentially reference several child records in another table.

***Many to Many Relationship*** - A relationship between tables in a database when a parent row in one table contains several child rows in the second table, and vice versa.

**2.1.0 Contributors**

Lauren Cox

Kylie Kent

Makayla Ballenger

Bri Cochran

Matt Strand

**3.1.0 File Descriptions**

***components***

* Folder that includes the bootstrap files.

***components/bootstrap2/css/bootstrap.css***

* Contains the css for the calendar page, specifically the buttons that control what you are viewing on the Calendar.

***css***

* Folder that contains the css for the calendar, specifically the background and line styles.

***css/calendar.css***

* Contains the css for the calendar itself.

***data***

* Folder that contains database connection and information.

***data/connect.php***

* Connects the database to the website.

***data/loggedin.php***

* Login connection for the website.

***data/logout.php***

* Logout connection for the website.

***data/teacheasy.sql***

* SQL file for the database for the website.

***fonts***

* Folder that contains the font for the logo.

***fonts/Eraser.ttf***

* Font for the TeachEasy logo.

***img***

* Folder that contains images for the calendar.

***img/dark\_wood.png***

* Image background when you select the date.

***includes***

* Folder that contains the header file that appears on every page.

***includes/header.php***

* Contains the structure for the header included on every page of the website.

***js***

* Folder that includes the JavaScript for the calendar

***js/app.js***

* Contains the JavaScript for the event Json files and the calendar views.

***js/calendar.js***

* Contains the general JavaScript for the calendar and events.

***js/calendar.min.js***

* Is the minimum required amount of JavaScript needed to run the Calendar.

***less***

* Folder that includes all the CSS building the Calendar.

***less/calendar.less***

* Contains imports of all the other less files.

***less/day.less***

* Is the CSS for the day event.

***less/events.less***

* Is the CSS for the event colors.

***less/grid.less***

* Is the CSS for the grid in the calendar.

***less/month.less***

* Is the CSS for the ticks on month.

***less/theme.less***

* Is the CSS for the year.

***less/variables.less***

* Is the variables used in the other less files.

***less/week.less***

* Is the CSS for the week view.

***resources***

* Folder that contains the image used on the login screen.

***resources/chalkBoard.jpg***

* Image used on the login screen.

***tmpls***

* Contains the html for the views the calendar can use and connecting them all together.

***tmpls/day.html***

* Contains the html and Javascript for the times in day.

***tmpls/event-list.html***

* Contains html and JavaScript for the event list and dropdown from the events.

***tmpls/modal.html***

* Contains the html and JavaScript for the start and end of the calendar.

***tmpls/month.html***

* Contains the html and JavaScript for setting the days in the calendar.

***tmpls/month-day.html***

* Contains the html and JavaScript to connect the day and the month.

***tmpls/week.html***

* Contains the html and JavaScript for all the weeks a calendar can have.

***tmpls/week-days.html***

* Contains the html and JavaScript for connecting the weeks to the days.

***tmpls/year.html***

* Contains the html for the year view.

***tmpls/year-month.html***

* Contains the html and Javascript for connecting the year to the months.

***.bowerrc***

* Directory to the bootstrap files in components.

***.gitignore***

* It allows github to share the ignore rules with any other users that clone the repository.

***Absences.php***

* Contains the php for the take attendance page in the main page of attendance.

***Attendance.php***

* Contains the php for the main page on attendance.

***Bower.json***

* Is used for the management of the front end.

***Calendar.php***

* Contains the webpage for the calendar.

***Composer.json***

* Contains a list of composer json objects with additional dist and/or source information.

***English.php***

* Contains for the php used in the english tab under gradebook

.

***Event.json.php***

* Used to add, edit or remove events from the calendar.

***Gruntfile.js***

* Contains JavaScript that connects the less files and tasks.

***Header.php***

* Contains the header navigation bar at the top of every page.

***Index.php***

* Contains the html for the first page that the website pulls up.

***lessonPlans.php***

* Contains the webpage for the lesson plan that displays on November 1st.

***lessonPlans2.php***

* Contains the webpage for the lesson plan that displays on November 20th.

***LICENSE***

* The license for the calendar package.

***Math.php***

* Contains the php for the math section under gradebook.

***Package.json***

* Connects the bower and gruntfile for bootstrap use.

***README.md***

* General information about the directory.

***Science.php***

* Contains the php for the science tab under gradebook.

***Socialstudies.php***

* Contains the php for the social studies tab under gradebook.

***teachEasyStyle.css***

* Contains the style for all pages of the website except for the calendar page.

***Web.config***

* Contains configuration settings that affect all Web applications on a server

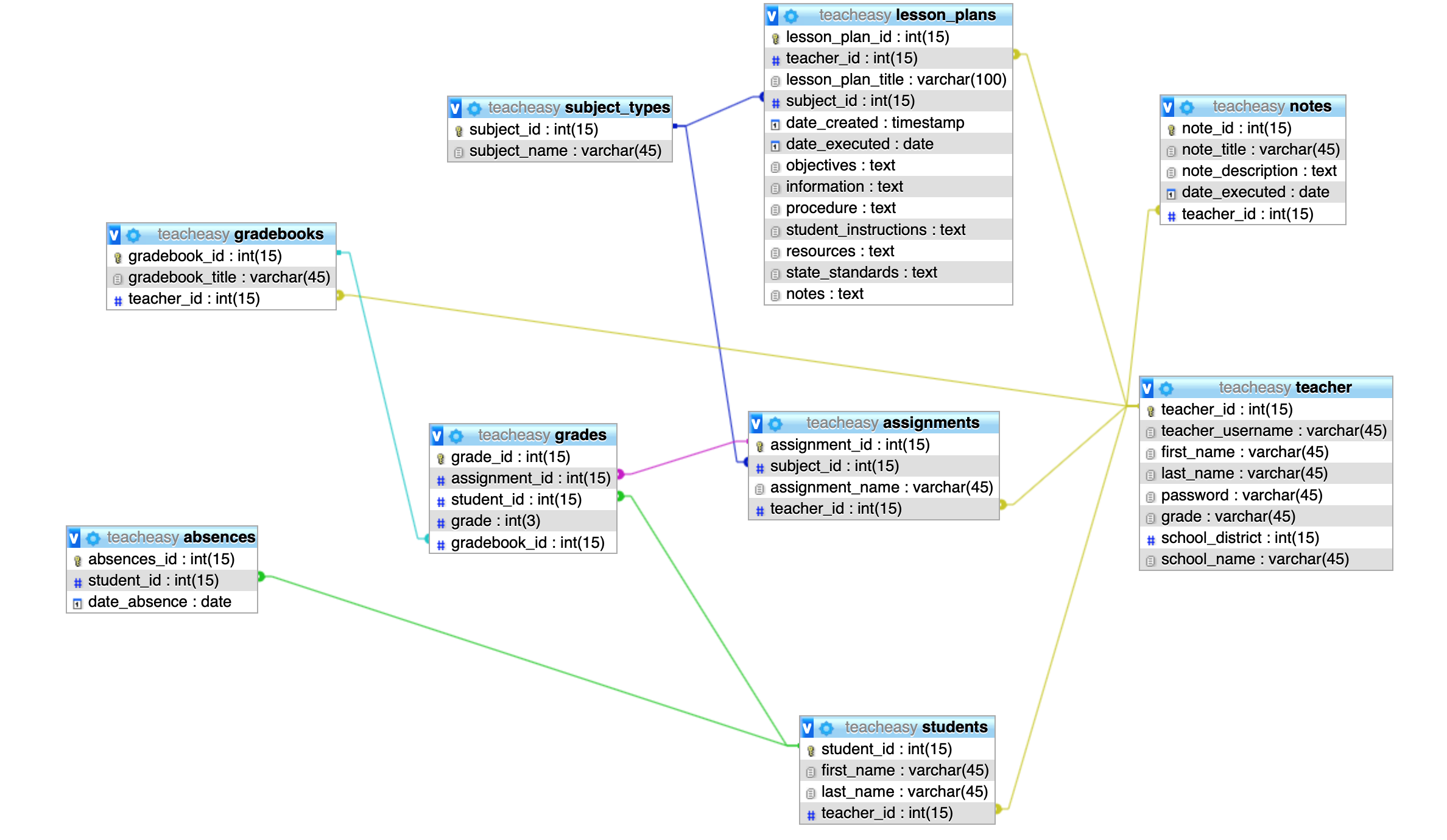
**4.1.0 Database**

**4.1.1 Definition**

A database is a collection of information/data that is stored in a table and is easily accessible, manageable, and updated. The TeachEasy database stores teachers' (any elementary education teacher in West Virginia) classroom information such as lesson plans, notes/ reminders, absences, and grades. Figure 4.1.2 is an ER model of the TeachEasy database which shows the relationship between tables and what type of information that is stored in the database. The ER model displays the character type of the information stored. For example, in table notes the column note\_description is a text type. The ER model also shows which columns are primary keys by displaying a gold key beside the column name.

**4.1.2 Tables**

**Figure 4.1.2**

****

* **Teacher**

The teacher table stores information about the teacher such as their first name, last name, username, password, etc. The Primary Key in the teacher table is teacher\_id. Each teacher has their unique teacher\_id which is used to identify which classroom information belongs to what teacher. The teacher table has five one-to-many relationships with the tables notes, lesson\_plans, students, assignments, and gradebooks. For example, a teacher can have/create multiple lesson plans, but a lesson plan only has one teacher.

* **Students**

The students table stores information about students such as their first name and last name. The Primary Key in the students table is student\_id which allows one to uniquely identifies each student. The students table has a Foreign Key teacher\_id which allows one to map which students belong to what teacher/classroom. The students table has three one-to-many relationships with the tables teacher, absences, and grades. For example, a student can have multiple grades, but a grade belongs to only one student.

* **Lesson Plans**

The lesson plans table stores information about West Virginia state standards lesson plan requirements such as procedure, objective, resources, etc. The Primary key in the lesson plans table is lesson\_plan\_id and the Foreign Keys are teacher\_id and subject\_id. The Foreign Key teacher\_id allows one to map which lesson plan belongs to what teacher and the subject\_id lets one know which subject the lesson plan cover (ex: Mathematics, Language Arts, etc). The lesson plans table has two one-to-many relationship with the tables teacher and subject type.

* **Notes**

The notes table stores information about the teacher's notes such as title, description, and date executed. The notes table Primary Key is note\_id and has one Foreign Key teacher\_id. The Foreign Key teacher\_id allows one to map which notes belong to what teacher. The notes table only has one one-to-many relationship with the teacher table. For example, a teacher can create many notes, but note only has one teacher.

* **Gradebooks**

The Gradebooks table stores the gradebook title, the Primary Key gradebook\_id and a Foreign Key teacher\_id. The Foreign Key teacher\_id allows one to map which gradebooks belongs to what teacher. The gradebooks table has two one-to-many relationships with the tables grades and teacher. For example, a gradebook can hold multiple grades, but a grade belongs only to one grade book. A teacher can have multiple gradebooks, but a gradebook belong to one teacher.

* **Assignments**

The assignments table contains the Primary Key assignment\_id, assignment name, and two Foreign Keys subject\_id and teacher\_id. The Foreign Key subject\_id lets one know which subject the assignment covers and the tearcher\_id show which teacher created the assignment. The assignments table has three one-to-many relationships with the tables teacher, subject types and grades. For example, a teacher can create multiple assignments, but an assignment belongs to only one teacher.

* **Subject Types**

The subject types table only contains the Primary Key subject\_id and subject name. The subject types table has two one-to-many relationships with the tables assignments and lesson plans. An assignment can only have one subject, but a subject type can have multiple assignments.

* **Grades**

The grades table contains the grade of the students. The grades table Primary Key is grade\_id and has three Foreign Keys which are assignment\_id, student\_id and gradebook\_id. The Foreign key assignment\_id allows one to map which grades belong to what assignments and the key student\_id show what student got that grade. The other Foreign Key gradebook\_id let one map which grades belong to what gradebook. The grades table as three one-to-many relationships with the tables gradebooks, students, and assignment. For example, a student can have multiple grades, but a grade belongs to a student.

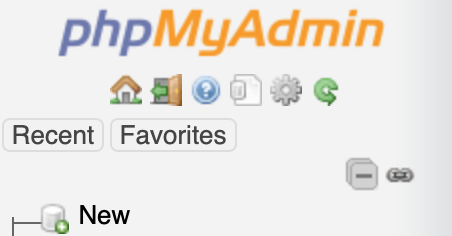
* **Absences**

The absences table contains the information on what days the student(s) missed school. The table's Primary Key is absences\_id. The absences table has one Foreign Key student\_id which allows one map what days the students missed. The absences table has one one-to-many relationship with the students table.

**4.1.3 Instructions/Implementing**

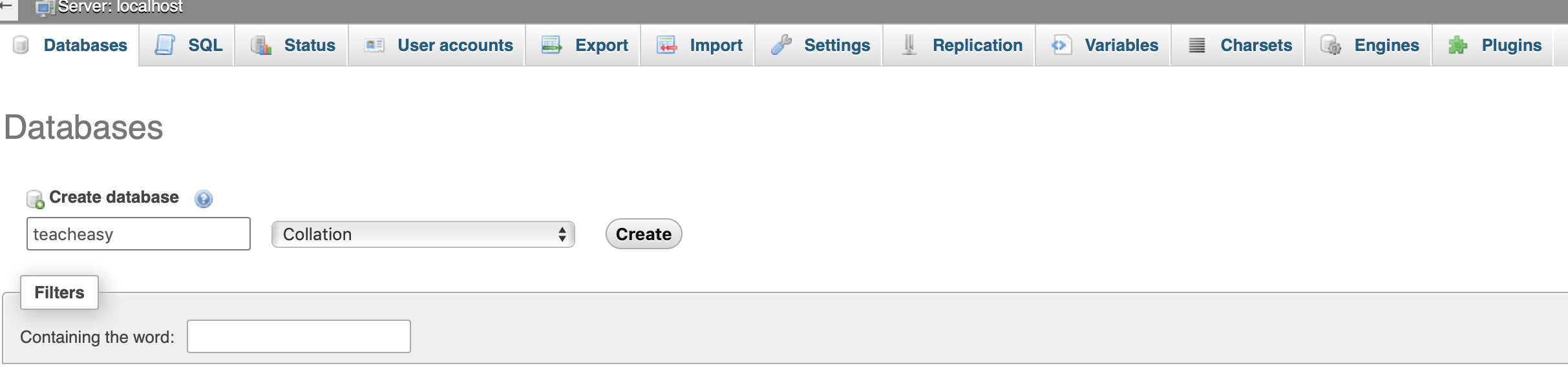
* Once one has XAMPP downloaded and installed on a computer, then hit the XAMPP application icon and start Apache and the MySQL. Open the internet browser on the computer and type in the address bar localhost then press enter/return. In the top right corner hit PhpMyAdmin. The computer should load a new page.

**Figure 4.1.3.0**

****

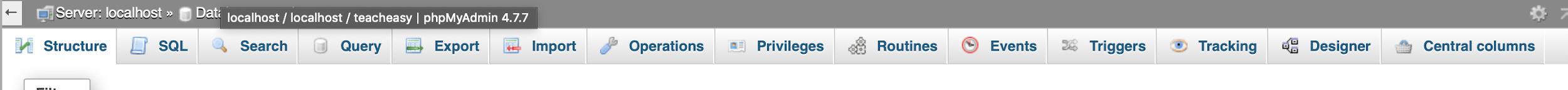
* In the left corner hit New with the green plus sign (figure 4.1.3.0)

**Figure 4.1.3.1**

****

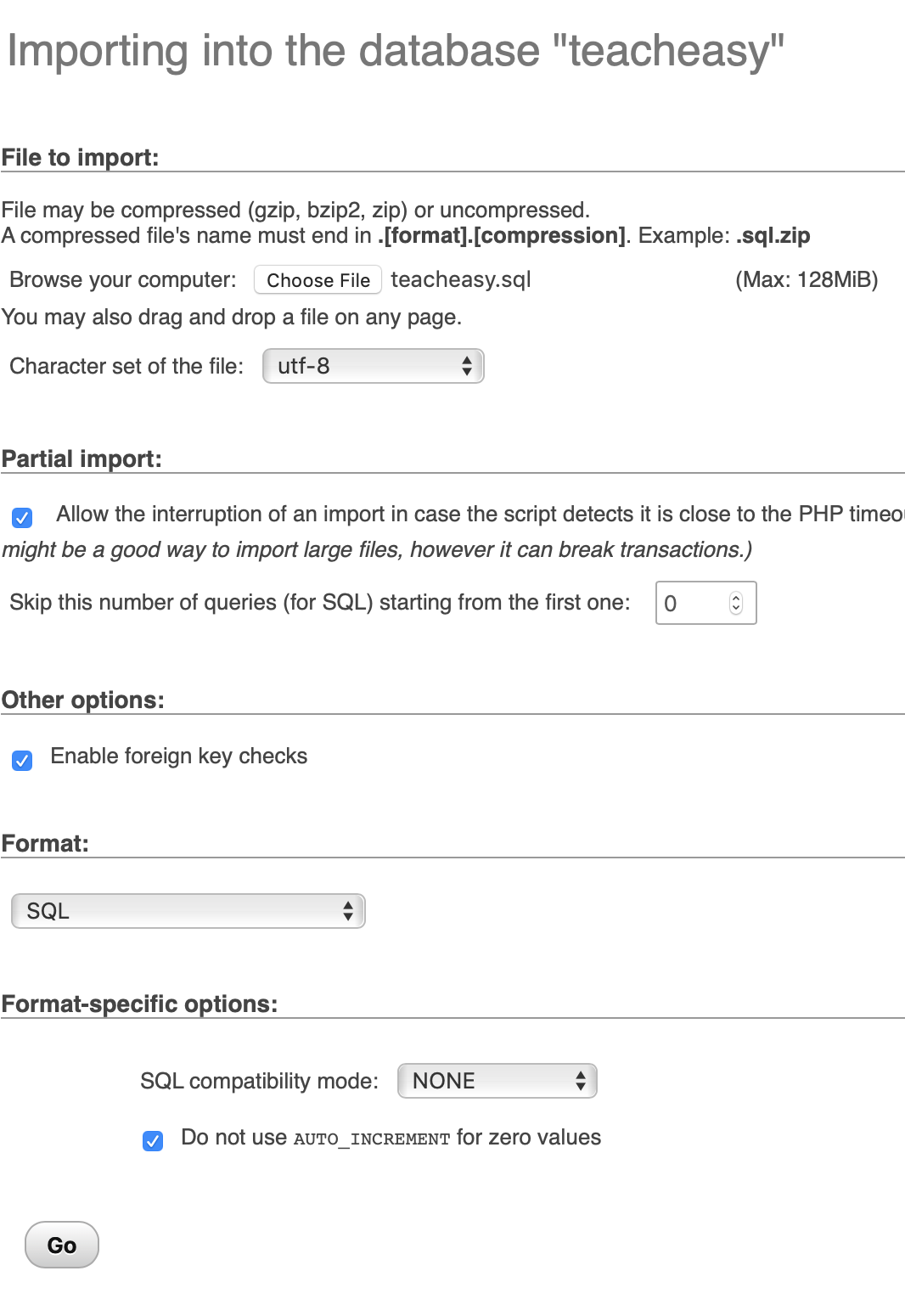
* Type in “teacheasy” under Create database and hit the button create (figure 4.1.3.1).

**Figure 4.1.3.2**



* Hit teacheasy on the left side, then up at the top of the webpage hit the import button (figure 4.1.3.2).

**Figure 4.1.3.3**

****

* Hit choose file and select the file teacheasy.sql. After you select your file hit go and the data was import into PhpMyAdmin.

**5.1.0 Calendar**

The calendar portion only includes the calendar, the different views of the calendar, and the events that appear on the calendar.

**5.1.1 License**

The Calendar is a package from a user on GitHub. The link to the GitHub repository <https://github.com/serhioromano/bootstrap-calendar>.

**5.1.2 Bootstrap**

The style of the calendar mostly uses Bootstrap, refer to the files CSS and Components for the base style of the calendar.

**5.1.3 JavaScript**

The Calendar uses JavaScript for the functionality of the calendar itself, refer to the file js.

**5.1.4 Events**

Events contain two major parts. Adding the event and clicking on an event. Adding events adds the dot and the title to the calendar with the event being clickable. Clicking on event displays the lesson Plan that the event represents.

* **Adding Events**

Events added to the calendar are done manually and are connected to the calendar by a json format,refer to the events.json.php for where they are added.

* **Clicking an Event**

Clicking on an event opens a new page that displays the events information which is a lesson plan directly from a database. Refer to lessonPlan.php for the php that displays the lesson plan.

**5.1.5 Lesson Plan**

The lesson plan uses PHP and is connected to the database via a single sql statement. If connection to the event buttons is desired jQuery and Ajax must be used to send Javascript variables to the lesson plan php file.

**6.1.0 Attendance**

This page is intended to be used for the manipulation of the attendance of each student based on the logged in user.

**6.1.1 Dependencies**

* includes/header.php

**6.1.2 Data Tables**

* **Attendance Tab Splash Page**

The table will display the student’s id, name, total absences, and the dates they were absent.

**6.1.3 Queries**

* **Attendance Tab-Main Page**

This table is generated by a query that ***SELECT***s the student’s ID, First and Last Name, and the dates they were absent.

***COUNT*** is then taken of the total results for the number of absences for each student. These are the columns which are displayed on the page.

To obtain these fields which are selected, two tables’ data must be ***JOIN***ed. The data is related by the s*tudents ID(student\_id)*.

The ***WHERE*** clause in this statement ensures that the students are selected based on the logged in teacher which is stored as a SESSION variable *‘userId’.*

***GROUP BY*** is used to ensure that the absences are for each individual student as per their ID.

***ORDER*** is selected by the students last name in alphabetical order.

* **Attendance Tab-Take Attendance**

A list of students is generated from the teacher. WHERE the student’s id, and name are related to the SESSION variable (userId) which contains the teacher’s Id.

* **Attendance Tab-Take Attendance, Insert Absences**

The absences table contains both the student’s id, and the dates they were absent, so both values can be directly inserted using the date that is given at the top of the page.

* **Attendance Tab-Take Attendance, Insert Absences Check if it already exists**

This query SELECTs the student, and their absences are then checked using the result to be sure the absence doesn’t already exist before adding the new absence.

* **Attendance Tab-Remove Single Absence Modal**

This query performs a simple SQL DELETE function on an absence on a specific student in the absences table.

* **Attendance Tab-Add Single Absence Modal**

The first query in this modal SELECTs the student and theirs absences, an if statement then checks if the query has any results and if it does, an error message is returned. If there is no result, the record is added to the absences table based on the student id.

**6.1.4 Modals**

Modals are used to eliminate reloading pages and to save time for the user, they take simple data input and handle small tasks.

**6.1.5 Take Attendance Page**

This page can be used for taking the entire class’ attendance. There is a way to modify the date, which automatically sets to today’s date, but it can be changed in case the user forgot to take attendance or wants to add future absences. The user MUST hit submit once the checkboxes have been selected, otherwise the data will not be saved to the database.

**7.1.0 Gradebook**

The gradebook is used to display the logged in teacher’s class based on the selected subject. It is also used to add new assignments and grades.

**7.1.1 Table**

The table is generated with two queries, the first generates the assignment names WHERE the teacher Id and subject Id match. The second generates the students names and grades for each assignment, at the end of each row, a new box is added so the user has the option of adding a new assignment.

**7.1.2 Grade Entry**

Grade entry is accomplished in the modal that appears upon clicking the button labeled “Add New Assignment”. The request cannot be submitted without having all the boxes filled. Missing assignments should be entered as “0”.

**7.1.3 Queries**

* **Displaying Grades**

The first query is used to SELECT the names of the assignments from the assignments table.

The second query, which displays each student and their grades, is don by JOINing 3 tables, assignments, teachers, and students. Each record returned has the students name and grade attached to it, but the name is only displayed once.

* **Adding New Assignments**

To add new assignments, a modal is brought up that lists each students name and an input box where the grade percentage is entered.

These students are generated from a query that selects each student based on the teacher\_id and orders them by last name.

Once the form is submitted, with all input boxes containing valid input, the data is sent to the database in 2 queries, one of which inserts the new assignment name into the assignments table, and the second inserts the grades with their corresponding student.

**8.1.0 Website**

We currently run our website using the XAMPP application. This can be downloaded simply by searching the web for XAMPP and following the install instructions.

We are assuming that you have access to our files from Github. Choose to “Clone or Download” the files and then choose “Download ZIP”. This will download the files to your device, and then you need to extract the files. When you do this, Extract All Files to your C: > xampp > htdocs folder. Once you are in the htdocs folder, take note of the name of the folder. Currently it would be “Software-Engineering-master”, but you can rename the folder to “teacheasy” for simplicity.

To view the website, launch XAMPP and Start the Apache and the MySQL modules. Once these modules are running, open a new web browser and type localhost/teacheasy (or whatever the file name is). This will launch the web page we have created for our Teach Easy software.