CS348 – Project – Stage 1

Due Date: 1/20/2024 at 11:59 pm

Introduction to the CS348 Project

In this semester, you will develop a database-backed web or mobile application. The goal of this project is to use course concepts in a real application. The project will allow you to practice most of the following concepts:

- Database design
  - Indexing.
- A database query language (most likely SQL).
  - Stored procedures in databases.
- Using a database query language in code (e.g., using SQL in Python or Java).
- Transactions and concurrency.
- Cloud databases

The project will follow a self-learning approach. You will need to choose and learn about a programming language and a web/mobile framework/stack to develop your application. The project evaluation will focus on the back-end of the application (especially code and SQL used to access the database). Therefore, **any graphical user interface will be accepted** as long as a regular user can utilize the features of your application.

Sample Application

This is a sample application that presents example database and features. For your project, you will choose a different application.

This application serves student clubs management and regular students. Student clubs organize different types of meetings. Meeting organizers will be able to create a meeting and to invite students to the meeting. Students will be able to RSVP (yes, no, maybe). The system provides reports, such as a list of attendees for a specific meeting and statistics regarding the meetings in a specific period of time.

Note that you only need to develop two main features in the application as described below.

Database Design (sample data organized in tables):

Students(student_id, name, email)
Meetings(id, date, time, duration, description, club_id, room_id, invitedCount, acceptedCount)
MeetingOrganizers(meeting_id, student_id)
Clubs(id, name, address, description)
Rooms (id, building, number, maxCapicity)

Requirements

Your application (in stage 2 and stage 3) should support two main features:

1. **Requirement 1:** An interface that allows users to add, edit, and delete data in one main table. This may also require adding, editing, or deleting data in other supporting tables. For example, in the previous application you can develop a page for creating, editing, and deleting a meeting. The page allows the user to choose/change students to be the meeting organizers, which results in adding/editing/deleting rows in the MeetingOrganizers table. If you choose to develop such interface for meetings then you do NOT have to develop interfaces to support add, edit, and delete operations for other tables, such as students, rooms, and clubs.

2. **Requirement 2:** One report interface that allows a user to select which data to display in a report. For example, in the previous application you can develop an interface that allows a user to filter meetings by date (From a start date to an end date), club, and room. Given rooms and clubs are stored in the database, the page retrieves those to build the user interface (e.g., a drop-down list that contains all rooms). After the user picks the room, club, and date range, the application generates a report of the matching meetings and some statistics, such as the average duration time, average number of invited students, average number of accepted invitations, and average attendance rate (attended/invited).

Stage 1 deliverables:

In this stage, you will need to:

1. Select a web/mobile programming language and framework/stack and install it in your computer. Start learning about this language/framework if you have not used it before.
2. Develop a sample web page (or an equivalent mobile window). The web page can include anything!
3. Record a demo of 1 to 2 minutes showing that you have installed the programming language/framework and you have developed a web page.

Stage 1 Submission:

Submit your demo to Brightspace/Content/Project/stage1

Web framework/stacks options:

Some of the frameworks/stacks used by students in the last semester:

- Python, Flask, SQL Alchemy, React
- Python, Django
- MERN stack (MongoDB, Express.js, React, and Node.js)
- SolidStart/SolidJS, Prisma
- Swift/UIKit/SQLite.swift
- Angular, MySQL, Spring Boot

CS348 – Project – Stages 2 and 3

Stage 2 Due Date: 3/26/2024

Stage 3 Due Date: 4/26/2024

Stage 2 deliverables:

1. Database design (e.g., database tables, primary keys, and foreign keys for a relational database).
2. Sketches of the user interface of Requirements 1 and 2 (from stage 1), including the report generated by Req. 2. You can use PowerPoint to draw how a page or report will look like when you finish development. If you have already developed a requirement, then a screenshot of the interface is sufficient.
3. A 5-minutes demo for Requirement 1 or 2 (described in stage 1).
    a. The user interface of requirements 1 or 2 must retrieve data from the database if needed, such as when populating the items of a drop-down list. For example, a drop-down list or a list of check-boxes to select a particular course. Such a list must be built dynamically using data from the database. The courses must not be hard-coded in the interface. In your demo, show how you build those interface components dynamically.
    b. In your project, use at least two of the following methods: **prepared statements**, **ORM**, and **stored procedures**. Each method should account for at least 20% of your database-access code (e.g., 80% prepared statements and 20% ORM). You may consider using ORM for data entry and updates (Req 1) and use prepared statements for your report (Req 2).

Stage 2 Grading Rubric

Stage 2 is 30% of the project grade

Deliverable	% of project grade
Database design	5%
Sketches of the user interface	5%
Demo. A proof that requirements 1 or 2 are complete. For example, in Req 1, data can be inserted, updated, and deleted. Also, the data can be displayed after the insert, update, and delete. Remember to highlight the source code while showing/explaining the above features.	20%
Total	30%

Stage 3 deliverables:

1. A document to describe what indexes you have on your tables and what query(s) and report(s) those indexes support. For each index, list the query(s) that benefit from the index and where the queries are used (e.g., a specific report). Note that indexes will be covered in the next few lectures.
2. A final demo of your project. You will record a 5- to 15-minute demo. In the demo, **show requirements 1 and 2 (described in stage 1) and how to use them**. In the demo, **describe parts of your code where you implemented the following course concepts**:
    a. Using at least two of the following methods: **prepared statements**, **ORM**, and **stored procedures**. Each method should account for at least 20% of your database-access code (e.g., 80% prepared statements and 20% ORM). You may consider using ORM for data entry and updates as well as simple queries and use prepared statements for complex reports.
    b. The user interface of requirements 1 and 2 must retrieve data from the database if needed, such as when populating the items of a drop-down list (e.g., a drop-down list or a list of check-boxes to select a particular course. Such a list must be built dynamically using data from the database. The courses must not be hard-coded in the interface). In your demo, show how you build those interface components dynamically.
    c. Transactions and your choice of isolation levels (transactions and concurrency will be covered in a few weeks). If your application is designed for a single user you may discuss a version where multiple users can access the same data concurrently.
    d. Discuss the lessons learned during the project phases. What would you change if you could start over.
3. A url to your application (if available) and your application's code (e.g., a GitHub link).

Extra credit:

1% of the course grade if you deploy your project (**database and application**) to Google Cloud Platform or any other cloud provider (e.g., Amazon AWS or Microsoft Azure). If you want to deploy your code to GCP you may use the app engine or compute engine (I think app engine is easier). If you do so, we expect a url to be available at the time of grading stage 3 (between 11:59 pm on 4/26/2024 and 11:59 pm on 5/4/2024). Remember to delete your instances after this date.

Stage 3 Grading Rubric

Deliverable	% of project grade
Indexes report. List queries that benefits from each index (be specific: report XXX or feat XXX). The indexes must be meaningful (important queries, reports, etc.). You must have a valid justification.	5%

Demo. A proof that requirements 1 and 2 are complete (data can be inserted, updated, and deleted). A report is displayed before and after some data is changed. Note. Remember to highlight the source code while showing/explaining the above features.	30%
Used two database access methods	10%
User interface built with data from the DB. If not applicable, 5% is added to 'using two access methods'	5%
Discussion of transactions, concurrent access to data, and the choice of isolation levels. If not applicable, 5% is added to indexes and 5% is added to demo.	5%
Discussion of lessons learned	3%
Copy of the code	2%
Total	60%
Url to the application. A test of the live application (data is entered, updated, or deleted and a report reflects the changes made by the TA). The demo proves the database and app are hosted by some cloud provider.	Extra credit (1% of the course grade).

AFAQ: Answers to Frequently Asked Questions

1. You may contact your TA to ask for help. However, the instructor and TAs will not be able to assist you in coding. You can ask for help by posting a question on Ed. Multiple students using the same tools are welcome to help one another by answering questions on Campuswire.
2. Can I use any database system? You are encouraged to use a relational database system (because we study those in more detail). However, you can use any database system that supports a query language, transactions, and indexes. stored procedures (or provides equivalent features). Popular options for databases that you can use are: MySQL, Postgres, Oracle, SQL Server, MongoDB (document JSON database), and SQLite.
3. Some students' MySQL instances in Google Cloud got hacked before. Do your research and follow security best practices in the cloud provider you are using (especially if you select to do the extra credit).
4. MongoDB users: Stored procedures are not available in MongoDB (stored javascript function are similar, but they are deprecated). Instead of stored procedures, you may try ORM for MongoDB (https://www.mongodb.com/developer/products/mongodb/mongodb-orms-odms-libraries/
5. Acknowledging minor bugs is encouraged. This is something you can discuss in the demo as items for lessons learned or future work.

6. If prepared statements are not supported in your language/framework find another secure method for protecting your application against SQL injection attacks. If no methods are available, sanitize the user inputs and explain your approach in doing so.
7. Between stages 2 and 3, you are allowed to change your database/UI design and/or the language/database/frameworks you are using. Discuss those changes in your final demo in stage 3.
8. There is no minimum number of indexes. However, make sure to support the important queries with indexes if needed, such as your report, your login (if you have one), retrieving data for your interface (e.g., populating data items for a drop-down menu or a list of check boxes). Those are frequent queries and must be supported with indexes.