## Project Description

Students in teams of 3 are required to select an application that would benefit from a database, and to build a database application from start to finish.

## Philosophy

Building a full database application from scratch allows you to control the process; instead of having the pieces decided for you, you must make all the decisions by yourself. Part of this process is that you will see how design decisions made at the beginning will affect your final project. We'll practice of some of the skills that you'll need, in the tutorials, and this will allow you to get some in-depth practice and a feel for what it's like when you make all the decisions for an application.

## Expectations of Group Members

Each group member must contribute meaningfully to the project. Each group member will be expected to participate in the demo, be thoroughly familiar with the application, and be able to answer questions from the TA. You will be required to use version control for your project.

In most cases, group members will share the same project grade; however, the instructors and TAs reserve the right to change the grade for each participant based on a student's contributions. For example, students who tend to be unavailable to their team, or who contribute little to the project, cannot expect to receive a good grade, even if the rest of the group gets a good grade. Note: Even if one of your group members volunteers to do most of the work for you, you still need to do your fair share.

Many groups tend to be fine, but if some member(s) of your group aren't carrying their fair share of the workload, contact your project group's TA as soon as you can. Your TA will try to get all members back on track. Don't wait until late in the project to address these problems! The earlier you come to us, the more likely it is that we can help you in a meaningful way.

We know that working with people in groups is not easy, but we believe that this is the right training for most IT and database jobs, going forward. So, this is valuable experience! Please work professionally. Like in the work world, not everybody is going to be the nicest person to work with, but you can still have a very productive working relationship. Having said that, if your group is having serious issues/conflicts that cannot be resolved by your team, let your TA know ASAP, and if issues persist, talk to your instructor.

## Strategies to Work Together

- We recommend that you set up a weekly meeting time with your team. As the semester progresses, most of your classes will have an increased workload. Meeting regularly can become a forcing function that will require you to make incremental progress before seeing your teammates.

- Before each meeting, create an agenda to lay out what you wish to discuss. This will help you streamline your meeting to be as efficient as possible and will force you to determine the most urgent needs of the team. Meeting together without any purpose will not help progress.

- Create meeting minutes that clearly indicate what you have discussed, any decisions that have been made, and assign specific task items to each member. Be sure to include due date information for each task. In the next set of meeting minutes, indicate the ongoing status of each task.

  Be sure to give everyone access to the minutes promptly after the meeting. This can be done through committing the minutes to your repository, sending it out via email, etc.

  You could manage issues through a tool like Jira but the downside is that this will be one more website you have to keep track of.

- Writing a good and detailed set of meeting minutes serves two functions.

  One, it helps clarify expectations for each team member. You want to ensure that everyone is on the same page about what needs to be done and understands why their task is important.

  Two, if the teaching team must step in, you have a paper trail which shows a timeline and progression of events.

- Be honest with your team. If something is giving you trouble, ask for help! If something has come up which is impacting your work, let your project TA know.

- Build in buffers when it comes to your deadlines. There can be many things that happen unexpectedly (e.g., getting sick) so you want to give yourself some protection against that.

- Let your project TA or instructor know early if things are going wrong! There are more available options we can explore if we know about an issue early. If we find out about issues at the end of the semester, there is a limit to what we can do.

## TA Check-Ins and Updates

There will be an official check-in at the midway point between the due dates for milestone 2 and 4. All group members are expected to attend the official check-in unless COVID causes us to have to change this mechanism. If you miss the official check-in without a valid reason, you will be penalized according to the rules listed in the syllabus. The purpose of the check-in is to determine if a group is on track, and whether each group member is contributing an approximately equal share of the work. Your project TA will also check your repository commits to see how much work has been completed and by whom.

Your project TA may occasionally touch base with you outside of the official check-in times. Groups are also welcome to contact their TA with any questions or issues.

Your TAs will be communicating with your group using Zoom. Their email addresses can be found on Canvas. If your TA hasn't contacted you, but you hear that a lot of other groups have already with met their TA, then you should send your TA a reminder, via e-mail. Also, please let your TA know ASAP if you are having problem with a group member or if there are severe teamwork issues.

## Learning Goals

- Deciding on an application for which a database system would be required
- Modelling the domain of the application, and defining the application functionalities
- Designing and implementing the schema
- Populating the database (but this should not be the main focus of the project)
- Writing the code needed to embed the database system in an application that has a Graphical User Interface (GUI).

Focus on the database aspects and the design of the schemas, as opposed to spending extensive time on sample data and the GUI.

You can use JavaScript, PHP, or another platform to connect your implementation to a relational database management system. Background and setup information can be found at the end of this document. Note that we will only provide support for JavaScript and PHP.

## Schedule

There are several intermediate deadlines that you must meet to ensure a successful project.

- Read the instructions carefully to make sure that you don't miss anything.
- Start early. Don't wait for the few days before the deadline to get started.
- Ensure that the cover page is turned in with all the milestones. The template of the cover page is available on Canvas.

The project has the following milestones:

| Milestone | Due Date |
|---|---|
| Milestone 0: Form a Group | See Canvas |
| Milestone 1: Project Proposal and ER Diagrams | See Canvas |
| Milestone 2: Logical Design including Relational Schema, SQL DDL, and Normalization (including any updates to your ER Diagram); List of Tentative Queries (in English) | See Canvas |
| Milestone 3: Project Check-In | See Canvas |
| Milestone 4: Code Implementation | See Canvas |
| Milestone 5: Demos | See Canvas |
| Milestone 6: Individual and Peer Evaluations | See Canvas |

### Milestone Submission Rules

Refer to the syllabus for the rules surrounding milestone submissions.

## Talking to the TAs

After the "Milestone 0: Form a Group" phase, you will be assigned a project TA who will act as your project supervisor. If you have any questions about how to best proceed with your project, your project TA is your first point of contact. Note that your project TA will not pre-grade anything (i.e., they will not tell you what to put for an answer to achieve a particular grade, but they can point you in the right direction and help to clarify expectations).

# Background Information

## Project Platforms

You may use any legal platform that you like, as long as:

- The final application uses a relational DBMS to manage the data.
- You don't use an automated system to generate the code and SQL (we want you to write these on your own).
    - Please refer to Tutorials 5, 6, and 7 which will be posted on Canvas early for your use and help. These tutorials are project tutorials including the most common tech stack that 304 projects have used historically (e.g., Oracle and JavaScript and Oracle and PHP, along with some unsupported materials for things like MySQL and Java/JDBC).
- You do not use a GUI generating tool.
- You may not use a programming assistance tool (e.g., Github Co-pilot, ChatGPT, etc.).
- All other requirements are met. Ask, if in doubt.
- **You must use version control for your code. CPSC 304 will provision repositories for each team to use and you must periodically commit your code to this repository.**

We are providing support for Oracle from the CS machines, along with either JavaScript or PHP. Please note that if you choose to implement with anything other than the provided and recommended infrastructure (e.g., running PHP on your own server, Java/JDBC), you may be asked to submit additional information. Furthermore, no support for anything other than the recommended software will be provided. Note that problems with non-supported platforms will NOT be accepted as an excuse for late project submission. So, use them at your own risk!

Refer to the syllabus for rules around using generative AI.

You can decide on the platform later, but as a preview, you can refer to some project resources we have here.