Project Documentation Requirements

1. Project description (3-5 pages)

You must describe your project and explain how all the entities interconnect to create your application. In

addition, you need to:

- Specify your group members and the entities they are responsible for.

- Include a list of features you implement and why these add value to your project.

- Include an installation manual, how to set up (database, folder structure, web alias, etc).

- Include a list with specific examples of how the technical concepts were implemented (see requirements)

- The database:

o Except if it is an associative table that breaks the M:N relationship, each table/entity in the database

should have at least five columns

o If you are in a team of two, the number of entities > count($members). One of the entities can be an

associative entity

o If you are in a team of three, the number of entities > count($members). One of the entities must be an associative entity

- Entities must be stored in a Database and accessed using PDO

There are two entities in the database, see more at Database details below.

User Table which contains the various

- The application must support CRUD (Create, Read, Update and Delete) Items using PDO. This must be

implemented for each entity. It does not mean that you should have the same list view, add view and edit view

for each entity. It should be a complete web application, not a data view and entry application.

- The application must have multiple HTML forms and pages.

- The user must be able to search and lookup for records from the database.

- Statistics must be shown for at least one entity.

- All input must be validated, and the appropriate events handled; proper English must be used.

- The app must be easy to use and visually pleasing, professionally designed with effective use of layouts, text,

page and content elements and user input elements.

- Users must be able to login; their credentials must be encrypted. The application must use sessions logging in

and logging out.

- The above requirements are for passing the project. You should try your best to make your project stand out

by integrating new technologies or topic that are not covered in the class.

- Include a list of the technologies implemented as an extension beyond what was covered in the classroom.

2. Class diagram

Specify the entities, the PK and FK and cardinalities. You can use tools like Software Ideas Modeller or any other

UML drawing tools like lucidchart.com to create your model

3. Meeting minutes

For those working in a group, every time the group meets, be sure to record the meeting minutes. Be sure to

include who was present and provide details on:

- Attendance (who is present for your meeting)

- What was done since the last time the group met (per person)?

- What is working well?

- What is not working well?

- What will be done before the next time the team meets?

Course: CSIS3280-001 BACKEND WEB DEVELOPMENT

Logistics:

Links:

Latest meeting note:

<https://docs.google.com/document/d/1CPxe6M_CGIk4EhlHPLl1iZhRxJfs_5yUqt5mI81Nvvc/edit?usp=sharing>

Google Drive:

<https://drive.google.com/drive/folders/1idI_rNAA2-JcdJI2lJoRg3FRLYv7vzqO?usp=sharing>

Github:

<https://github.com/WCEdison/CSIS3280_PHP_Project.git>

Personal info:

Leader: Manj, Dilraj 300331319

Member: Edison CHAN 300370372

1. Reconfirm key design feature

1. Format: Store front
2. Major tech decision
   1. Use Laravel
   2. Use Github to publish webpage
   3. Find Food/Grocery API
   4. Use local database
3. Design direction
   1. Purpose:
      1. Grocery Store
   2. Features:
      1. Login -> As manager
      2. Customer
         1. View
         2. Filter (Basically Sql select, use desc/asec, sort by)
      3. Management (use from last demo)
         1. Add
         2. Edit
         3. Delete
   3. Database
      1. 2 x Unrelated Tables
      2. Admin login
      3. Product
4. Question to ask: How much weight is this project and when is the deadline
   1. If high then we add more feature (Might expand if scope expands)
   2. Otherwise we keep current feature + grocery api + login for admin + database editing for admin

2. Logistics: How to communicate, Github project and publishing

1. Whatsapp to do normal communication
   1. Error
   2. To dos
2. Teams for bigger meeting → On demand only (one more handing it in)
3. Github + Github desktop
4. Use [Visio] for Class Diagram

---------------------------------------------

3. Task to be done

1. Criteria Grading The web application produces errors or warning messages. The output of var\_dump or print\_r are visible on the web app. The web app creates folders and files automatically. -16 **(QA Both of us)**
2. Project submitted and named properly with all assets to Blackboard by Team Leader, file is named according to the naming convention 1 point **(Dil)**
3. Project Description, installation manual, Class Diagram, Meeting Minutes, all completed with relevant details. Class Diagram **(Wait til classes are designed)**  is accurate and includes all properties and cardinality. User Manual – Concise, to the point, graphical, labelled properly. 3 points **(Ed)**
4. Good program structure is used, all functions were described with comments, and comments are used where applicable. Naming conventions are followed. 2 points

**(Both of us) - list of function and class with comments**

1. The database was designed following the requirements. The tables/entities were sound and were designed to reflect real life web application. The complexity of the database should be similar to or greater than any database design used in the class practice. The SQL script was included in the project submission. 3 points **(Dil)**
2. CRUD operations are implemented using PDO, DAO and prepared statements. The errors are captured and logged. The user must be given positive confirmation of a CRUD operation. Statistics are coded using SQL queries and displayed. 5 points **(Dil)**
3. All input were validated according to the expected input. The user must be prompted for corrections and stack traces must be logged to a file. 2 points **(Ed)**
4. HTML Forms were coded properly. The layout is efficient and easy to use, and interface controls are intuitive. 2 points **(Both of us)**
5. The overall quality and complexity of the submitted project should be greater than any class demo practices and assignments 2 points **(Both of us)**
6. Bonus requirements 2 points **(Ed → Publish on Github)**

4. To do & milestone

1. Set github
2. Find a Grocery/Food API
3. Learning how to publish webpage github
4. Set up a database design
5. Set up database
6. Set up PDO and DAO
7. Set up validation
8. **Worst Case: Switch to CSS + HTML (if laravel doesn’t work) + Switch to React**

4. User manual (1 page)

Provide a concise user manual how to install, use and operate the application.

- Include visuals wherever applicable

- Must cover all the actions a user can perform (CRUD and Search etc...)