**Project Status and Design Report**

|  |  |  |
| --- | --- | --- |
| **Topic:** | *CLC – Final Project: Completed E-commerce Database based Web Application* | |
| **Date:** | *12/02/2018* | |
| **Revision:** | *1.7* | |
| **Team:** | 1. *Fredrick Ondieki* | |
| 1. Michael Weaver | |
|  | |
|  | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | | *Fredrick’s Work* | *Fredrick* | *6+* |  | | *Documentation* | *Michael* | *2* |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **GIT URL:** | *https://github.com/FREDDYSMALLZ/E-Commerce* | |
|  | *https://github.com/battousairurik/E-Commerce* | |
| **Peer Review:** | *Y/N* | Fredrick needs to edit to reflect his participation |

Project Presentation Links

Presentation Link – Michael’s Part: <https://www.youtube.com/watch?v=7Gx20jWrVYI&feature=youtu.be>

Presentation Link – Fredrick’s Part:

Documentation Update Log

1. General Technical Approach
2. Technical Issue: At current the program relies of having an active local database. For future revisions, a cloud database would need to be utilized for an active website. Utilizing a site such as Hostable would be optimal.
3. Code Improvement: The Admin Index Page was refactored to act as a controller, calling an include on the page that is requested by a particular call. This idea can prove to be efficient in optimizing program flow and simplifying page calls. This is typical in MVC architecture projects.
4. Mobile Access: The project interface is not currently programmed to work with mobile devices. The webpages would need code to dynamically resize based on the mobile device being utilized. A mobile app would need to be developed to optimize the application efficiency.
5. Agile Retrospective Results
6. Any last minute changes will be reflected in the application homepage at GitHub

**Planning Documentation**

**Agile Retrospective Results:**

*The following table should be completed after each Retrospective on Things That Went Well (Keep Doing). An alternative to the following table is to use a Mind Mapping tool such as Coggle. If you use a Mind Mapping tool you must include a URL or Image File.*

|  |
| --- |
| **What Went Well** |
| **The new partial page implementation of the project was a great idea on Fredricks part. The more pages that are partial and can be handled with includes, the more code reusability the project has.** |
|  |
|  |

*The following table should be completed after each Retrospective on Things That Didn’t Go Well (Stop Doing) and What Would Be Done Differently Next Time with an Action Plan to Improve (Try Doing and Continuous Improvement). An alternative to the following table is to use a Mind Mapping tool such as Coggle. If you use a Mind Mapping tool you must include a URL or Image File.*

|  |  |  |
| --- | --- | --- |
| **What Did Not Go Well** | **Action Plan** | **Due Date** |
| **Local Database requirements** | **Utilize an online database connection or one built into the project** |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Design Documentation**

**Install Instructions:**

*Step by step instructions for setting up your database, configuring, and deploying/installing your application. This section should also include detailed instructions for what configuration files are required by your application, what configuration settings need to be adjusted for various runtime (development or production) environments, and where the files need to be deployed to. This section should also contain detailed instructions for how to clone your application source code from BitBucket and deploy the application to an externally hosted site.*

1. Clone the GitHub repository

2. Launch with PhpStorm

\*\* Hosting options and additional details will be included with future updates

**General Technical Approach:**

*You should, in words, describe your approach and design here. You should also summarize any meeting notes, brainstorming sessions, etc. that you want to retain thru the design of your project.*

*Proper Project Layout*

All PHP files are organized according to their type, Navigation, Partial Pages, Admin, Public, etc. Proper file layout prevents pages from being access incorrectly or development from being confusing. The Admin index isn’t easily mistaken for the public index.

*Separate Navigation Bars*

There are separate navigation bars between the Admin section and the main website, each with links to their respective page lists. Navigation bars are handled as a partial page and included in all necessary files to optimize code reusability.

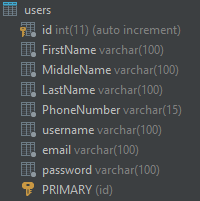
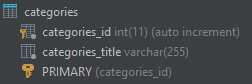
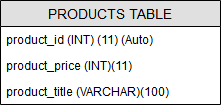
**Key Technical Design Decisions:**

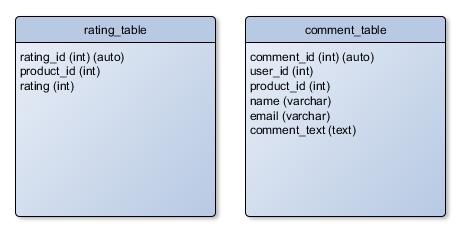
*Any final technical design decisions, such as framework decisions, etc., should be documented here. This should list the technology/framework, its purpose in the design, and why it was chosen.*

*Key technical approaches include, the use of a header, footer, and include statements to optimize code reusability.*

**ER Diagram:**

*Image file of your ER database diagram.*

*  *

**

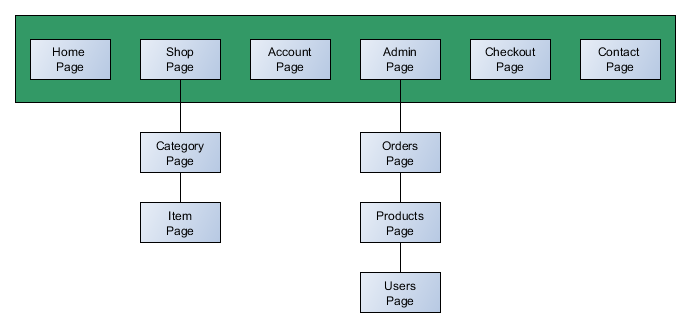
**DDL Scripts:**

*This should contain a link to BitBucket where the DDL script can be downloaded from.*

Not included as of version 1.2

**Sitemap Diagram:**

*Image file of your Sitemap diagram.*

**

**Security Design:**

*This section should outline the design for how authentication and authorization was supported. This section should also contain all of the roles and privileges that are supported by the design.*

*Security includes input validation and login credentials.*

**Third Part Interface Design:**

*This section should fully document any Third Party Service Interface API’s, how to access the service, what parameters are required by the API, and the detailed JSON data format specification that could be used by a third party developer to integrate with the service and API.*

*There is no third-party support as of version 1.2*

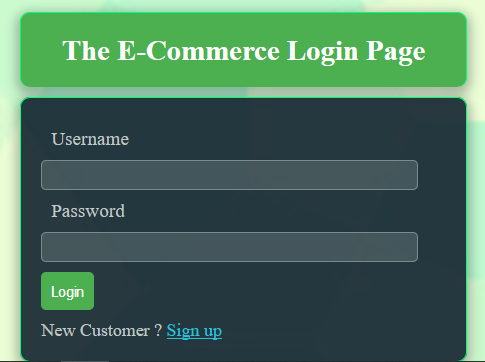
**Flow Charts:**

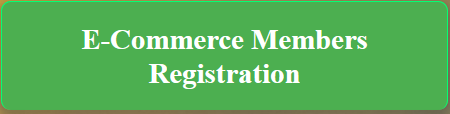
*You should insert any flow charts here. Flow charts should document algorithms or workflow that will be implemented in your program. At a minimum this should contain a flow chart of the Minesweeper game logic.*

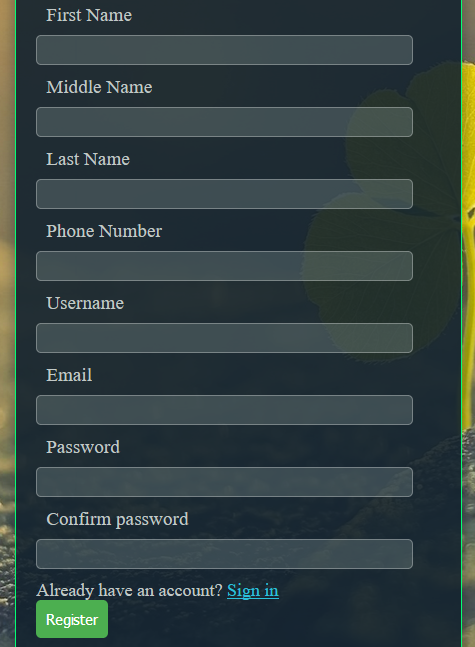
Not Included as of version 1.2

**User Interface Diagrams:**

*You should insert any wireframe drawings or white board concepts that were developed to support your application. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*







**Class Diagrams:**

*You should insert any class diagrams here. Your class diagrams should be drawn correctly with the three appropriate class compartments, + and – minus to indicate accessibility, and the data types for the state/properties as well as method arguments and return types. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*

Not included as of version 1.2

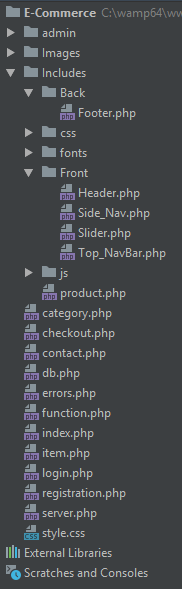
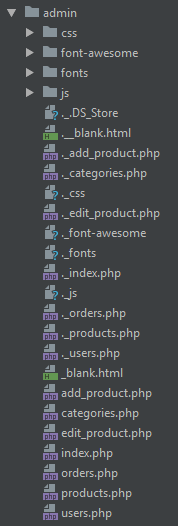
**Pseudo Code:**

*You should provide BitBucket URL references to any code stubs & pseudo code. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*

Not included as of version 1.2

**Other Documentation:**

*You should insert any additional drawings, storyboards, white board pictures, project schedules, tasks lists, etc. that support your approach, design, and project. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*

* *

**Results of SCRUM Retrospective:**

Not included as of version 1.2