Design Document

Team Nova

December 2, 2017

SE 305

Jacob Gordon, Jonathan Hedman, Mark Kirshenman, Nathanial Fleet

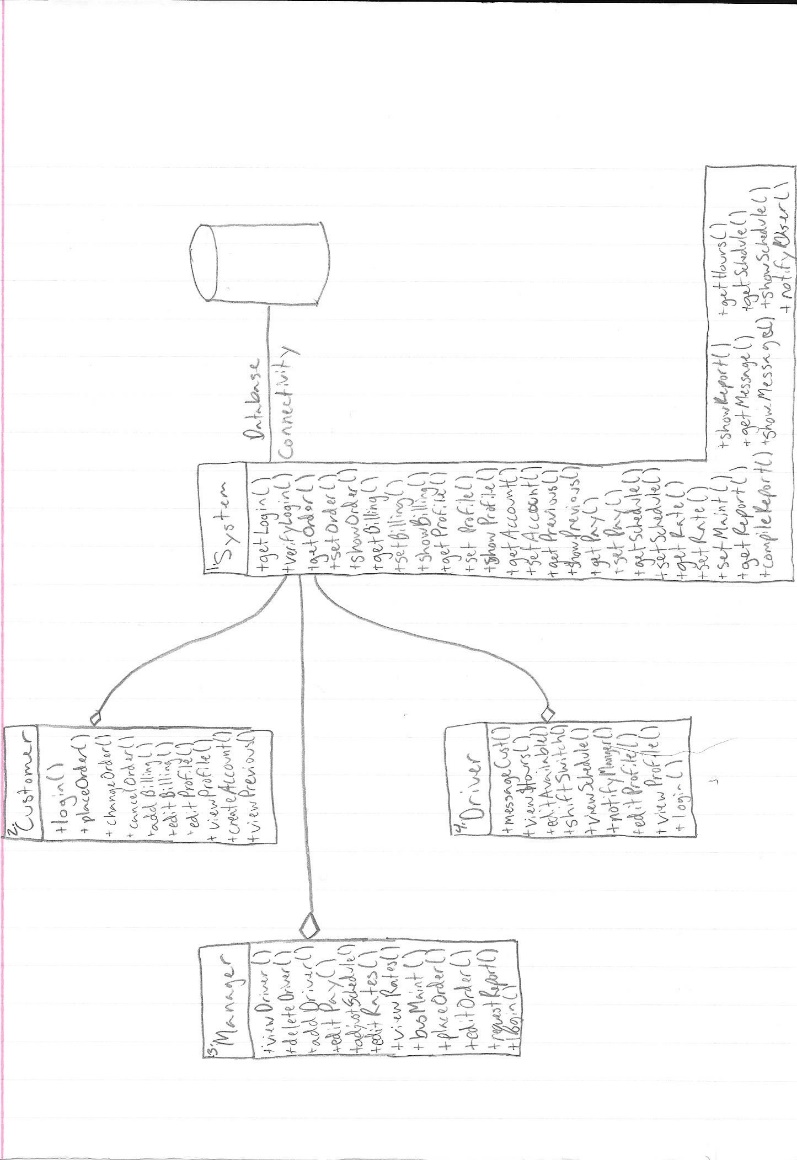
Table of Index

1. High-level architectural design
   1. Two proposed high-level designs for system
   2. Address the system attributes
2. Medium-level design
   1. Design of the subsystems
   2. User interface design
   3. Design of the database
   4. Work Breakdown Structure

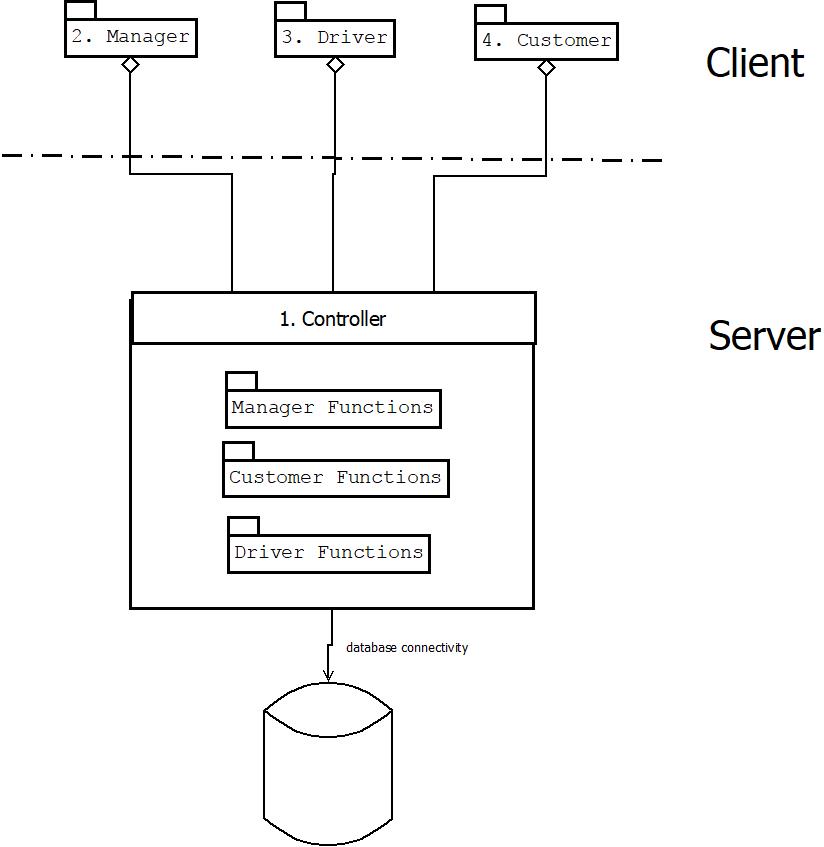
Appendix A

1. High-level architectural design Written by Mark, Proofread by Jacob
   1. Two proposed high-level designs for system

Repository Written by all



Client Server Written by all



* 1. Address the system attributes Written by Jacob, proofread by Nate

1. System: This subroutine contains every function that the system is reliable for performing to ensure the program runs as intended.
2. Customer: This subroutine lists every functionality the customer will have access to within the program.
3. Manager: Within this subroutine is listed all the features of the manager to perform their required tasks.
4. Driver: This subroutine has all of the functionalities each driver will be able to use inside of the program.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute | Priority | Repository |  | Client Server |  |
| reliable | 5 | 3 | 15 | 4 | 20 |
| availability | 4 | 4 | 16 | 4 | 16 |
| security | 3 | 1 | 3 | 2 | 6 |
| modifiability | 2 | 4 | 8 | 5 | 10 |
| portability | 1 | 4 | 4 | 3 | 3 |
|  |  |  | 46 |  | 55 |

Reliability-

We thought both of these designs were very reliable. For Repository we can easily identify all the required functions and make sure nothing is missing from for the system. Being able to make sure we have all the functions necessary will make sure that it is reliable. For Main Program Subroutine it might be easier to miss certain functions accidently.

Availability-

We came to similar conclusions for availability on both diagrams as we did for their reliability. Each lays out each function in each subroutine to aid with writing the final program and producing a final product that will meet availability expectations.

Security-

Neither diagram does a very good job of helping to measure how secure and safe the system will end up being.

Modifiability-

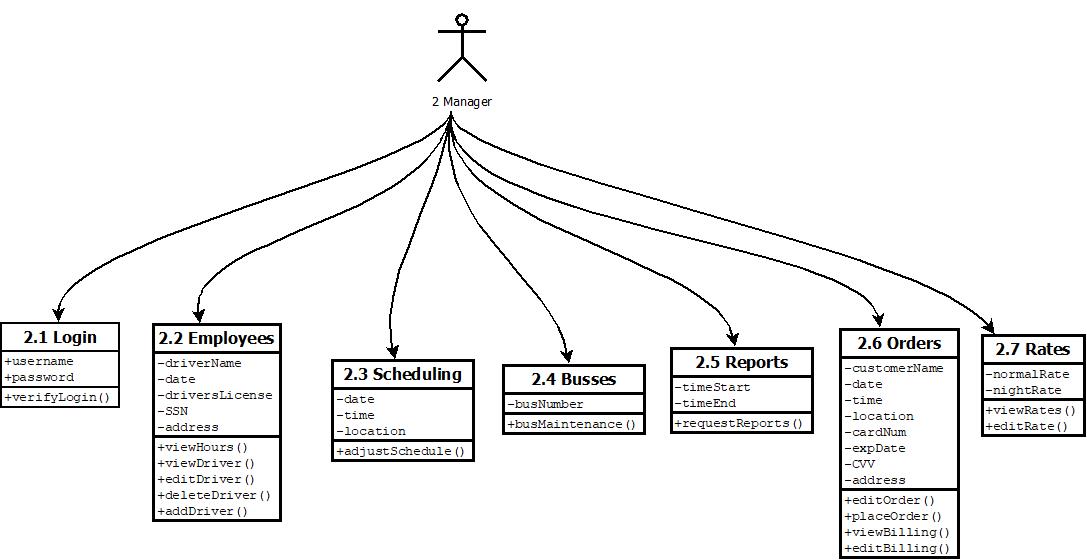
Repository is far easier to edit as all that needs to be done is add a function to the given subroutine rather than rearrange many subroutines in the Main Program diagram to make room for new ones.

Portability-

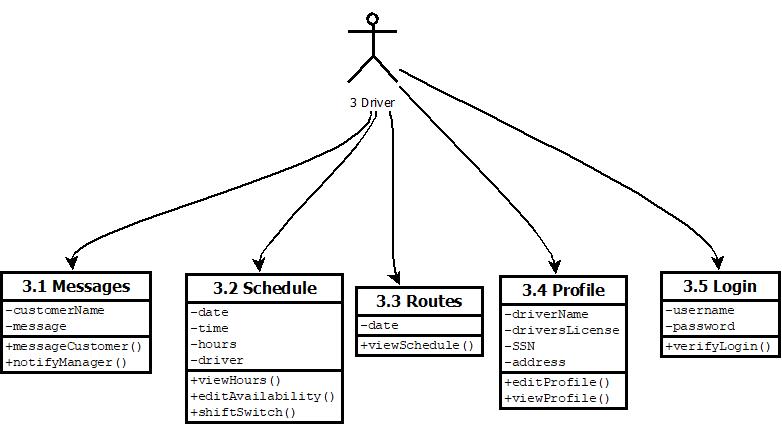
Each diagram is similar in its representation of portability as both show every function for each sub routine clearly to make coding and porting to other systems more manageable.

1. Medium-level design
   1. Design of the subsystems

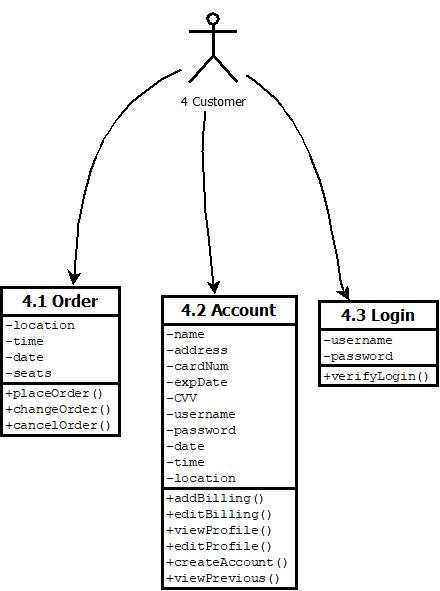
2.1.1 Manager Subsystem Written by Jacob



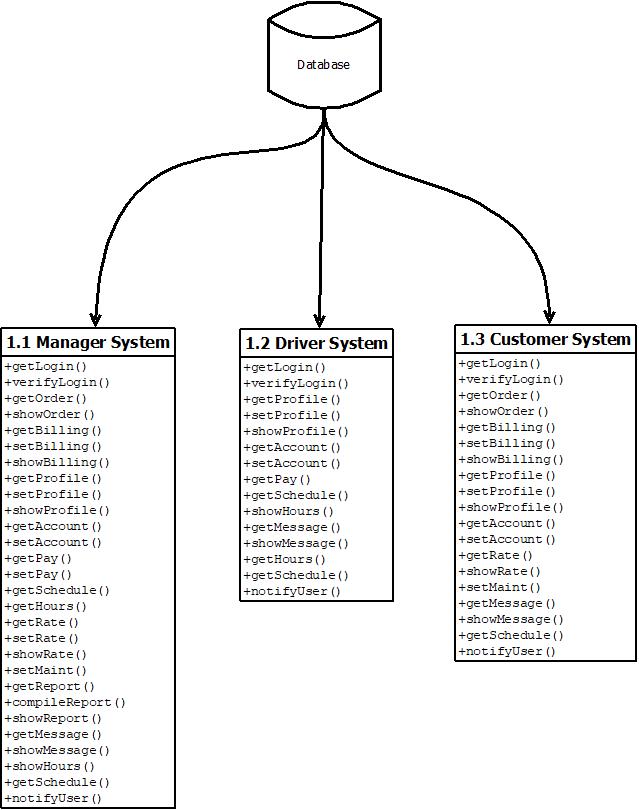
2.1.2 Driver Subsystem Written by Nate



2.1.3 Customer Subsystem Written by John



2.1.4 System Control Subsystem Written by Mark

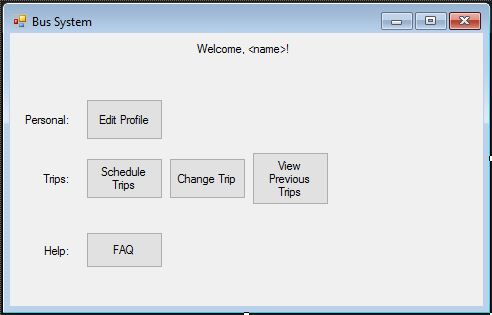


* 1. User interface design Written by John, proofread by all

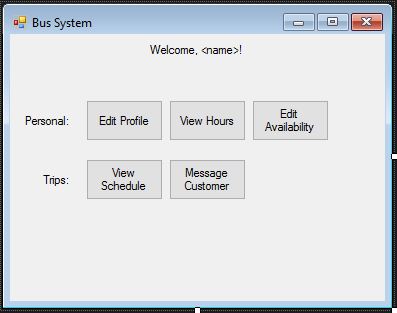
Login Page:



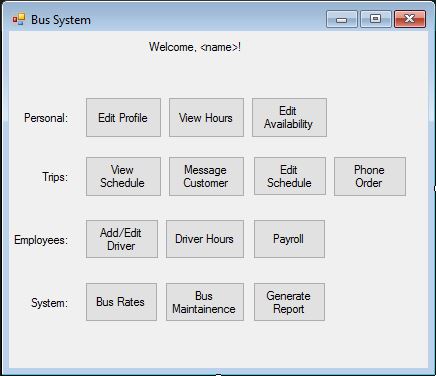
Customer Function page:



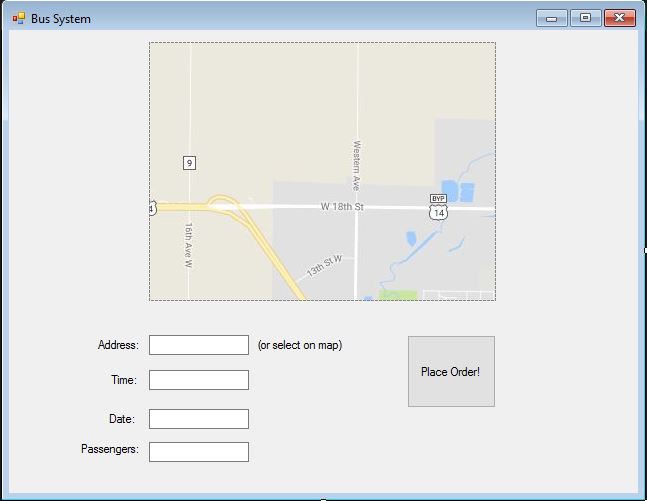
Driver Function page:



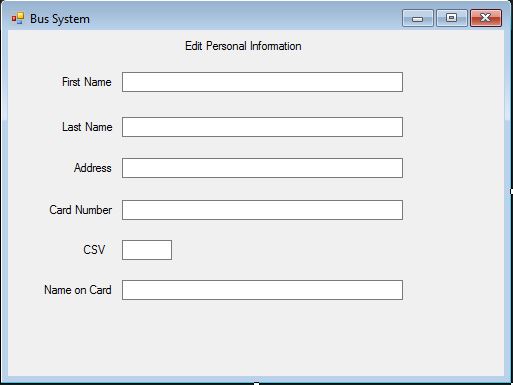
Manager Function page:



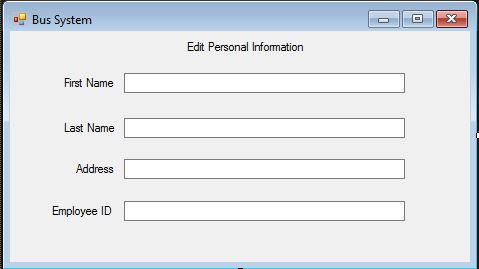
Place Order page:



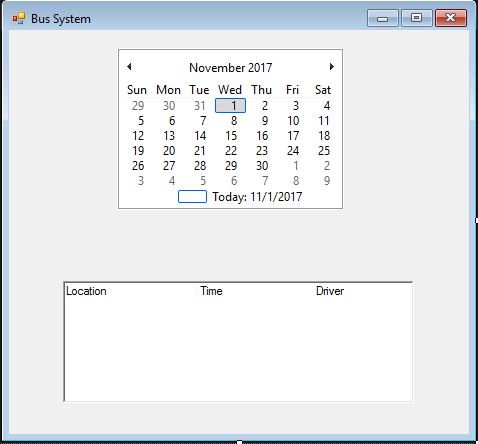
Edit personal information customer



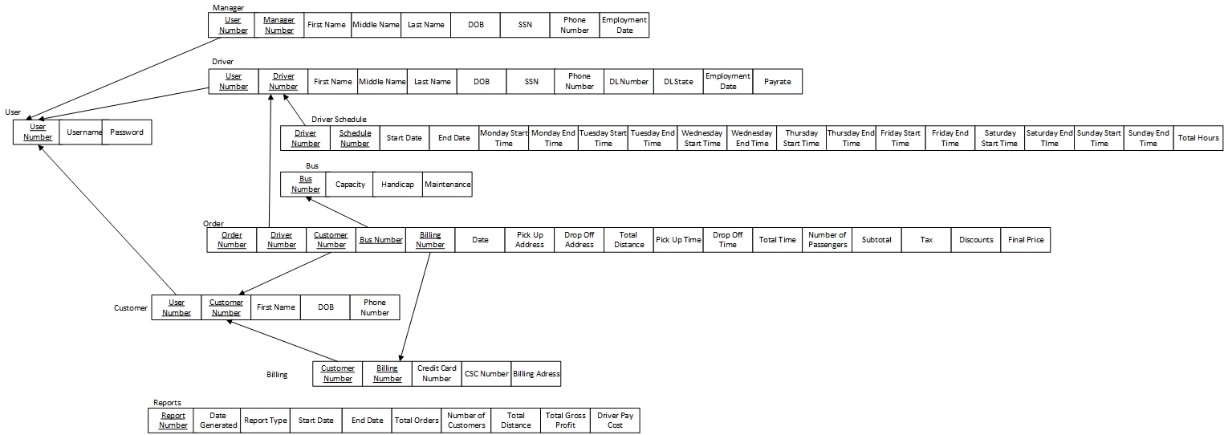
Edit personal information Driver



Schedule View



* 1. Design of the database Written by John, proofread by Jacob



* 1. Work Breakdown Structure Written by Nate, proofread by Mark

1. Software for Bus Order System
   1. Code function implementation
      1. Code Manager functions
      2. Code Driver functions
      3. Code Customer functions
      4. Code System functions
   2. Implement Database
      1. Design database tables
      2. Code database
      3. Link database to main code
      4. Implement user interface
   3. Test Software and database implementation
      1. Test software codes and functions
      2. Test database and code link implementation
   4. Write Documentation
      1. Complete final documentation
      2. Write help file

**Appendix A** Written by Mark, proofread by all

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Use case no.** | **Subsystem no.** | **Component no.** | **Activity no. in WBS** |
| verifyLogin | 2.1.1 | 2 | 2.1 | 1.1.4 |
| viewHours | 2.2.1 | 2 | 2.2 | 1.1.1 |
| viewDriver | 2.2.2 | 2 | 2.2 | 1.1.1 |
| editDriver | 2.2.3 | 2 | 2.2 | 1.1.1 |
| deleteDriver | 2.2.4 | 2 | 2.2 | 1.1.1 |
| addDriver | 2.2.5 | 2 | 2.2 | 1.1.1 |
| adjusSchedule | 2.3.1 | 2 | 2.3 | 1.1.1 |
| busMaintenance | 2.4.1 | 2 | 2.4 | 1.1.1 |
| requestReports | 2.5.1 | 2 | 2.5 | 1.1.1 |
| editOrder | 2.6.1 | 2 | 2.6 | 1.1.1 |
| placeOrder | 2.6.2 | 2 | 2.6 | 1.1.1 |
| viewBilling | 2.6.3 | 2 | 2.6 | 1.1.1 |
| editBilling | 2.6.4 | 2 | 2.6 | 1.1.1 |
| viewRates | 2.7.1 | 2 | 2.7 | 1.1.1 |
| editRates | 2.7.2 | 2 | 2.7 | 1.1.1 |
| messageCustomer | 3.1.1 | 3 | 3.1 | 1.1.2 |
| notifyManager | 3.1.2 | 3 | 3.1 | 1.1.2 |
| viewHours | 3.2.1 | 3 | 3.2 | 1.1.2 |
| editAvailability | 3.2.2 | 3 | 3.2 | 1.1.2 |
| shiftSwitch | 3.2.3 | 3 | 3.2 | 1.1.2 |
| viewSchedule | 3.3.1 | 3 | 3.3 | 1.1.2 |
| editProfile | 3.4.1 | 3 | 3.4 | 1.1.2 |
| viewProfile | 3.4.2 | 3 | 3.4 | 1.1.2 |
| verifyLogin | 3.5.1 | 3 | 3.5 | 1.1.4 |
| placeOrder | 4.1.1 | 4 | 4.1 | 1.1.3 |
| changeOrder | 4.1.2 | 4 | 4.1 | 1.1.3 |
| cancelOrder | 4.1.3 | 4 | 4.1 | 1.1.3 |
| addBilling | 4.2.1 | 4 | 4.2 | 1.1.3 |
| editBilling | 4.2.2 | 4 | 4.2 | 1.1.3 |
| viewProfile | 4.2.3 | 4 | 4.2 | 1.1.3 |
| editProfile | 4.2.4 | 4 | 4.2 | 1.1.3 |
| createAccount | 4.2.5 | 4 | 4.2 | 1.1.3 |
| viewPrevious | 4.2.6 | 4 | 4.2 | 1.1.3 |
| verifyLogin | 4.3.1 | 4 | 4.3 | 1.1.4 |