Team Middle Out

Project State

Version 1.0

December 5, 2016

Members

Charlton Smith [charltonuw@gmail.com]

Ameet Toor [ameet2r@uw.edu]

Karan Toor [toork@uw.edu]

Lovejit Hari [lovejitshari@gmail.com]

Braxton Rowe [rowebe@uw.edu]

Website

https://azoni.github.io/SoftwareDevelopment/

Introduction

In the beginning Middle Out set out to create an application to allow users to easily donate services or goods to those in need. The final deliverable shows the end product we hoped to accomplish based on the paper prototypes that were created for deliverable 1. There are a few functionalities that we wanted to implement but were not able to. For example, we wanted to allow the users to upload, and modify profile pictures, along with a few of the admin functionalities. We still use the admin class as an enum type that a user can potentially be promoted to in the future. The state of our final deliverable includes all of the core functionality we set out for. Our business rules are lacking a bit, since a lot of those situations will be handled on site, or in other areas outside of our software application.

When the program starts, there is a preset login. You may choose to continue on or sign up a new user. The login username and password verification is fully functional and will not let you login if you have not signed up. (The username for John Doe is john@doe.com, and password is password. Case sensitive!) John Doe is a dummy account that only saves changes during use of the program but does **not** save changes if you try to re-log as John Doe because it will go back to the preset values assigned to John Doe. To confirm if your information is actually there, you can play around and add services to a new user you have created, and watch as the pages update when changes are made. The changes you make to your own account are saved. Our application also has a map page that uses Dijkstra's algorithm to find the best route of travel from one area to another. We have selectable locations you can choose from and change and the map will update automatically. Drivers/Educators select locations where they are and the algorithm will find the best route to that location. A back button has also been added to pages so the user may go back at anytime. Finally, we have implemented a logout menu link to a page that confirms if you want to log out. Logging out will save user information from the previous log in of the same user and will also re-open the login form page.

Table of Contents

Requirements Addressed

User story 1:

Scenario about a driver who is interested in what our organization is about and how they can donate their time by picking-up/dropping-off goods

a. Business rules associated.
    i. BR122 Driver must provide identification on pickup, and donor must provide identification when his/her items are being picked up.
        1. We modified this business rule so that only the driver must provide identification upon signing up to drive. When they want to drive they must provide a picture of their license and insurance in one .jpg. And they must provide the type of vehicle, the capacity of the vehicle, and their availability start and end times.
        2. State of implementation:
            a. For this business rule we fully implemented the modified business rule. However, according to the original business rule, we did not fully implement the original rule specifications because we do not ask the donor for identification.
    ii. BR112 All volunteers must provide a current form of identification which will be kept on record.
        1. For this business rule we require a form of identification when the volunteer/user signs up as a driver. When they upload the identification, it is kept on record for that user. Each time they sign up for a new driving service, they have the option to change the identification and insurance photo.
    iii. BR115 Drivers who are moving perishable goods in refrigerators must do so in a specified time frame allocated by the donating body and corresponding with Center for Disease control regulations.
        1. For this business rule we changed it so the driver's are responsible for the goods.
        2. State of implementation: We have implemented a start and end availability for the driver. This is the time allotted to the driver and according to their location. The driver can use Dijkstra's algorithm mapping to find the quickest route to the location they're headed.
b. State of Implementation for user story 1:

i. For this user story we did not let the user choose if they were going to pick up or drop off goods. We made it so that the users service is just driving goods and the availability of the user. Our software can be used to determine availability via a phonecall by one of our staff members.
ii. We fully implemented this user story besides allowing the user to specify pick up or drop off.

User story 2:

Scenario about a good samaritan who is interested in donating books & clothes and would like to donate them to someone in need:

c. Business rules associated:
    i. BR120 We are not responsible for any damage(s) that may occur.
    ii. BR112 New donors must provide required documentation.
        1. For this business rule, we changed it so that they are already cleared on sign-up, also no photo ID requirement has been implemented for someone who donates items.
        2. State of implementation: We implemented the sign in and information saving, but we didn't require photo identification.
d. State of implementation for user story 2:
    i. For this user story, we have followed all functional requirements except for donors needing to provide a form of photo id. Also, we have shown a disclaimer stating that we aren't responsible for goods that are lost/stolen.
    ii. We haven't implemented a dynamic pick-up location, we do however have pre-set locations that a driver can go to-and-from, assuming donators drop off at donation centers.

User story 3:

Scenario where a person is looking to request some food.

e. Business rules associated:
    i. BR111 All recipients can receive aide a maximum of 3x/week
        1. For this business rule, we changed it so that it depends on when goods will arrive to the recipient. Also donation centers are usually where the goods will be delivered so they can monitor that and we're the middleman which helps donation centers/educators to people. Meaning we will not limit the number of times a user can receive goods, the centers will.

2. State of implementation: We haven't implemented it explicitly, it is assumed that the donation centers will handle this part. User information is stored on sign-up and whatever changes are made during the use of the program in regards to adding a service are also saved.

ii. BR122 Driver must provide identification on pickup, and person must provide identification when his/her items are being picked up.

1. For this business rule, we changed it so it depends on the donation center to require identification. This way the donation center mediates the exchange and our company is not held liable.
2. State of implementation: We haven't implemented this business rule aside from sign-up and how the information is changed during application use.

f. State of Implementation for user story 3:

i. We fully implemented this user story by allowing the user to create a request for goods, but the business rules were not implemented aside from small parts in the code.

How to run the application

To start the application unzip the file, then open the MiddleOut_Executable file. Once you are in the file, click on the MiddleOut.exe.

"John Doe" is an example and the user gets reset every time the program loads. The login information for John Doe is located in our introduction. Once you sign up a new user, you will notice whenever the program exits, if you sign in with the same username and password all of the services will be saved.

-- IMPORTANT
Combo boxes: Use the up and down arrow keys and press enter if the combo box doesn't allow you to select an option with a mouse. After you have made a selection, do not press enter after you have used arrow keys, simply click on another combo box and check if you can use the mouse to click on an option. You can continue on as normal for the rest of the program. This may be a Mac specific issue so selecting options from combo boxes should work fine on windows machines.

User Story 1:

The following script follows the user story scenario about a driver who is interested in what this organization is about and how they can donate their time by picking-up/dropping-off goods:

1) When you start the application, you will see the "Login" Screen. Click the button labeled "Sign up".
2) Enter in your information and click the button labeled "Sign up". ***Remember your email and password***
3) You will now be at the home screen. Click on the "Add a Service" option.
4) Here, you may select to Drive, Donate, Educate, or Request Goods. Select the "Driver" checkbox, then press "Continue".
5) On this screen, we require a photo of your Driver's License. Click the button labeled "Upload" (for purposes of this demo, you can go ahead and upload any picture from your computer as a .jpeg, .jpg, .png).
6) On the same screen, below the driver's license upload section, choose a car type, and a holding capacity(in milk crates), enter 20.
7) Below that there is an option to enter an availability range. The first box is for the start time, and the second box is for the end time. Enter a start and an end time i.e. 3:00 pm for the first box and 7:00 pm for the second box.
8) You will be taken to the thank you screen once you click on the "Finish" button and be prompted to return to the home screen. Assume that you have already been contacted and your information has been processed.
9) Click on "return to home"
10) On your home screen you will see the current service that you have associated with your account.
11) Click on "mapping" on the left-hand side to choose where you want to go. The idea was for the program to choose for you, but since we did not have enough time, we made it so that the user can choose. You can choose a start location and choose an end location. The yellow lines are all the paths you can go. Once you choose a start and end location, the shortest path will show in red.
12) Finally click on "about" to see what the application is about and who the developers are.
13) Close the application.

User Story 2:

The following script follows the user story scenario about a good samaritan who is interested in donating books & clothes and would like to donate them to someone in need:

1. After starting the program, enter in the information from **user story 1** and select sign in.
2. A new window should pop up, and you will be at the "Home" screen, and should be able to see a picture of our team and the services you made from the last user story.
3. From the left side, choose "Add A Service", the window you are currently in should change.
4. Select "Donate", then click the "Continue" button at the bottom of the window.
5. At this screen, you are given the option to choose one type of donation: Toys, Clothes, Tech, First Aid, Food, Hygiene Products, or Other. Select the "Other" checkbox, then type in the donation text box "books" and click the button labeled "Continue".
6. The window should change again, and you should see a page that says "Thank you!". Click the "Return to Home" button at the bottom of the page.
7. Now you will be back at the "Home" page, click on "Add A Service" on the left side of the window.
8. Choose "Donate", and then click the "Continue" button at the bottom of the page.
9. Choose "Clothes" and then click the "Continue" button at the bottom of the page.
10. You will be now see the "Thank you!" page, click the "Return to Home" button at the bottom of the window.
11. You will be back at the home page and will see the services that you currently have for your account.
12. Go to the profile button the left-hand side of the screen and you will see your information and your past service reports information.
13. Now you can press log out which is a menu link at the top right of the window, select the "yes" button which will take you back to the log-in screen.

User Story 3:

The following script follows the user story scenario where a person is looking to request some food.

1. After the program restarts, click on "sign up".
2. A new form will pop up and you can create a new user account by entering in all of the information in the fields.

3. Select sign up after you have entered your information.
4. A new window should pop up, and you will be at the “Home” screen, and should be able to see a picture of our team
5. From the left side, choose “Add A Service”, the window you are currently in should change.
6. Select “Request Goods”, then click the “Continue” button at the bottom of the window.
7. The window should change, now choose “Food”, then click the “Continue” button at the bottom of the window.
8. The window should change again, and you should see a page that says “Thank you!”. Click the “Return to Home” button at the bottom of the page.
9. You will be back at the home page and will see the services that you currently have for your account.
10. Now you can close the program by hitting the “X” in the top right of the window.

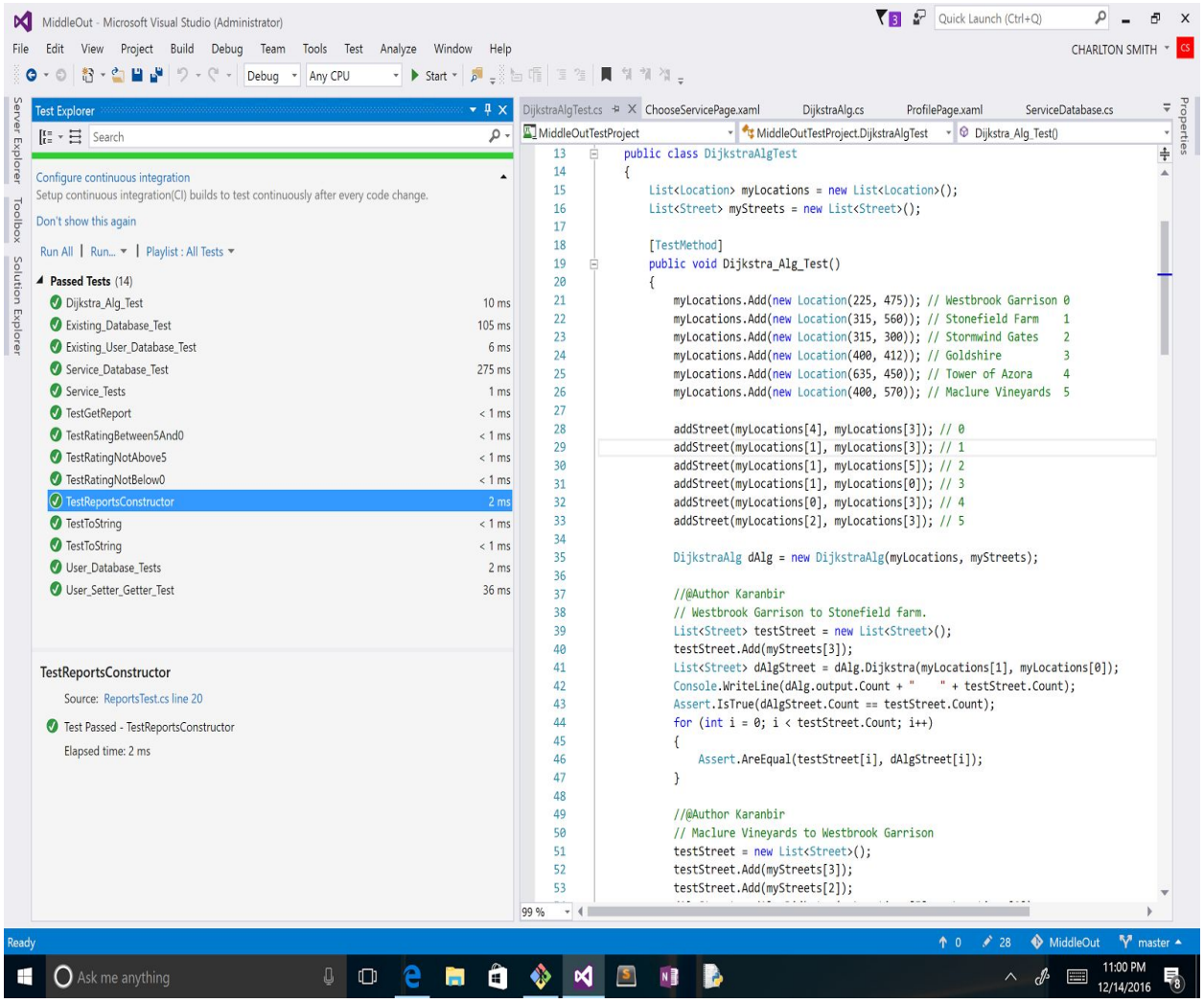
Contributions

Person	Class-Method	Activity
Charlton Smith	User.cs newUser.cs ThankYou.xaml.cs Mainwindow.xaml.cs AboutPage.xaml.cs Home.xaml.cs Home.xaml About.xaml ListPage1.xaml Information.xaml MainWindow.xaml DijkstraAlgTest.cs Form2.cs Form2.Designer.cs	Co-author Co-author Co-author Co-author Co-author Author Co-author Co-author Co-author Co-author Co-author Co-author Co-author Co-author
Karan Toor	MainWindow.xaml MainWindow.xaml.cs DijkstraAlg.cs Form1.cs Form1.Designer.cs	Co-author Co-author Author Co-author Co-author

	Form2.cs Form2.Designer.cs InformationPage.xaml InformationPage.xaml.cs Location.cs MapPage.xaml MapPage.xaml.cs Streets.cs	Co-author Co-author Co-author Co-author Author Author Author Author
Ameet Toor	ReportsTest.cs Reports.cs ReviewServiceTest.cs ChooseServicePage.xaml.cs EducatePage.xaml.cs InformationPage.xaml.cs MainWindow.xaml.cs RequestGoodsPage.xaml.cs ReviewService.cs	author Co-author Author Co-author Co-author Co-author Co-author Co-author Co-author
Lovejit Hari	About.xaml About.xaml.cs ProfilePage.xaml ProfilePage.xaml.cs ListPage1.xaml ListPage1.xaml.cs MainWindow.xaml LogOutPage.xaml LogOutPage.xaml.cs ThankYou.xaml ThankYou.xaml.cs	Co-Author Author Author Author Co-Author Co-Author Co-Author Author Co-Author Author Co-Author
Braxton Rowe	User.cs UserDatabase.cs Service.cs ServiceDatabase.cs UserTest.cs UserDataBaseTest.cs ServiceTest.cs ServiceDataBase.cs	Author Author Author Author Author Author Author Author

Tests

Pictures of Unit Tests:



How to run unit tests:

1. To run test codes, it is important that you **RUN VISUAL STUDIO AS ADMINISTRATOR!**
2. Then open the “MiddleOut” solution.

3. Once the project is open and fully loaded, on the menu bar at the top of Visual Studio click on “Test”.
4. Then “Run”, and then “All Tests”.

If there are Problems with unit testing:

If there are any problems and the tests do not run: clean solution and then rebuild solution, then go back and follow the directions to run unit tests.

To clean and rebuild the solution:

1. With Visual Studio open in administrator mode, and the project is loaded, click on the menu item called “Build”.
2. Then click “Clean Solution”, and wait for it to finish cleaning the solution; you will know it is finished when the Output shows “Clean: 2 succeeded, 0 failed, 0 skipped =======”.
3. Then click on the menu item called “Build”.
4. Then click “Rebuild Solution”, and wait for it to finish rebuilding the solution; you will know it is finished when the Output shows “Rebuild All: 2 succeeded, 0 failed, 0 skipped =======”.
5. Then follow the directions to run unit tests.

Source catalog

Because we did our project in C#, we do not have a .jar file. We tried putting our executable project into a file that has only the .dll files and the executable, but that did not work. The executable is called” MiddleOut.exe”, and is in the MiddeleOut>bin>debug>MiddleOut.exe.

In our source code the files are all in the MiddleOut file. The files and folders of importance will be listed below in a bullet fashion. Indented bullets mean the file or folder is inside the bullet that is its parent. Ex: MiddleOutTestProject is inside of the MiddleOut File.

- ProjectState.pdf
- MiddleOut file
    - MiddleOutTestProject
        - DijkstraAlgTest.cs
        - ReportsTest.cs

        - ReviewServiceTest.cs
        - ServiceDatabaseTest.cs
        - ServiceTest.cs
        - UesrDataBaseTest.cs
        - UserTest.cs
    - MiddleOut
        - Bin
            - Debug
                - MiddleOut.exe (This is our executable)
        - Images
            - MiddleOut.jpg
        - Pages
            - Home.xaml
            - Home.xaml.cs
            - ListPage1.xaml
            - ListPage1.xaml.cs
            - LogOutPage.xaml
            - LogOutPage.xaml.cs
            - SettingsPage.xaml
            - SettingsPage.xaml.cs
        - AboutPage.xaml
        - AboutPage.xaml.cs
        - App.xaml
        - App.xaml.cs
        - ChooseServicePage.xaml
        - ChooseServicePage.xaml.cs
        - DijkstraAlg.cs
        - EducatePage.xaml
        - EducatePage.xaml.cs
        - EnumTypes.cs
        - Form1.cs
        - Form1.Designer.cs
        - Form1.resx
        - Form2.cs
        - Form2.Designer.cs
        - Form2.resx
        - InformationPage.xaml
        - InformationPage.xaml.cs
        - Location.cs

- MainWindow.xaml
- MainWindow.xaml.cs
- MapPage.xaml
- MapPage.xaml.cs
- newUser.cs
- NotificationPage.xaml
- NotificationPage.xaml.cs
- ProfilePage.xaml
- ProfilePage.xaml.cs
- Reports.cs
- RequestGoodsPage.xaml
- RequestGoodsPage.xaml.cs
- ReviewService.cs
- Service.cs
- ServiceDatabase.cs
- Street.cs
- ThankYou.xaml
- ThankYou.xaml.cs
- User.cs
- UserDatabase.cs

Emergency contacts

Charlton Smith: charltonuw@gmail.com

Ameet Toor ameet2r@uw.edu, 208-841-6836

Karan Toor toork@uw.edu, 208-841-9167

Lovejit Hari: lovejitshari@gmail.com, 206-890-8416

Braxton Rowe rowebe@uw.edu, 253-278-9382