Sprint Dates: 4/23 to 4/29

Items to mention

Schedule

Member	4/23	4/24	4/25	4/26	4/27	4/28	4/29
Jason	2	1	4	1	4	2	1
Jesus	0	3	2	5	5	5	0
Vi	4	4	4	4	4	4	4
Gio	2	3	3	3	3	3	3
Rainier	0	3	3	3	0	2	3

Team Capacity

Members	Hours Available
Jason	15
Jesus	20
Vi	28
Gio	20
Rainier	14
Team Capacity (Total) =	97

Unavailable Days

Members	Unavailable Days	Reason
Jason		
Jesus	4/23, 4/29	Class
Vi		
Gio		
Rainier	4/23 4/27	Dental Appointment Work

Task Breakdown
Work Items Based on Product Backlog

Work Items	**Effort Points (Hours)**	**Assigned Owners**
Communication Establishment CE - 1	55	Gio
Communication Establishment CE - 2	55	Gio
Communication Establishment CE - 2	55	Gio
Inventory Management IM - 1	34	Rainier
Inventory Management IM - 2	34	Rainier
Car News Center Peer Review	2	Rainier
Car News Center CNC - 1	26	Vi
Car News Center CNC - 3	26	Vi
Vehicle Marketplace VPM - 1	16	Vi
Vehicle Marketplace VPM - 2	19	Vi
Vehicle Marketplace VPM - 3	20	Vi
Service Log SL - 1	40	Jason
Service Log SL - 2	40	Jason
Service Log SL - 3	40	Jason
Service Log SL - 4	40	Jason
Service Log SL - 5	60	Jason
User Administration UA - 1	21	Jason
User Administration UA - 2	21	Jason
User Administration UA - 3	21	Jason
Scrap Your Car Peer Review	2	Vi
Scrap Your Car SYC - 1	55	Jesus

Scrap Your Car SYC - 3	55	Jesus
Scrap Your Car SYC - 4	55	Jesus
Scrap Your Car SYC - 5	55	Jesus
Logging Web API	13	Jesus
Vehicle Profile VP - 1	34	Rainier
Vehicle Profile VP - 2	34	Rainier
Vehicle Profile VP - 3	34	Rainier
Vehicle Profile VP - 4	34	Rainier
Vehicle Profile VP - 5	34	Rainier

Donate Your Car

Donate Your Car (DYC - 1) (11)

1. Manager Layer (3)
   a. Create IManager interface for Donate Your Car
   b. The Manager Layer will use the function GetCharities that will use the ICharity to create an object of what it will get from the database
2. Service Layer (2)
   a. A RetrieveCharities function will be created that will return a collection of charities, descriptions, and links. Will use to
3. Front end (3)
   a. A page will be created that will display all the charities with their descriptions and a button will be created next to each charity that will hold the link to them
4. Entry point (3)
   a. GetCharities will send a HTTP Get to get charity names, description, and links from the database

Donate Your Car (DYC - 2) (10)

1. Manager Layer (3)
   a. The GetCarDetails manager layer will call an existing service from Vehicle Profile feature to get the details needed to fill out the charity forms.
2. Service Layer(1)
   a. Will get called from the manager layer and this will already be an existing service so no coding will be required
3. Front End (3)
   a. When the user clicks the button relevant to the charity it will create GetCarDetails that will return the necessary information to fill out the charity forms.

   b. Once the details are returned they are then redirected to the link that was in the button and a script will be used to fill out the online form
4. Entry Point (3)
   a. GetCarDetails will use a GET HTTP request and will use the IVehicleProfile to get the details needed to fill out the forms such as make, model, and year

__**Communication Establishment**__

- **Communication Establishment (CE-1)(20)**
  1. xUnit tests (3)
     a. Create a test for retrieving the correct seller from the database that is associated to the marketplace post by the VIN number.
     b. Create a test that checks if a chat session was created in the database when a request was sent to the seller
  2. Manager Layer (5)
     a. Create CommunicationEstalishmentManager where only the VIN will be sent to the service layer that will use the VIN as the search parameter for ISearchParameters service.
     b. Manager layer will be used again once it has the seller’s username and be used to create CreateChatSession that will upset the buyer's username, seller’s username, and VIN of the post.
  3. Models(1)
     a. Create the IChat interface that will be used by the services to create a chat session. Will contain values such as VIN, SessionID, BuyerUsername, SellerUsername, ViewStatus, MessageCreationTime and MessageContent
  4. Service Layer(3)
     a. The first service that be used is GetSellerInfo that will be in charged of retrieving the username of the seller from the database
     b. The second service will be CreateChatSession where it will create a new object called the IChat object that will be used to upload into the database
  5. Front End(5)
     a. Create a button that requests communication with the seller and will gather the VIN of the post and also the username of the buyer which will be the person who clicked the button.
  6. Entry Point(3)
     a. Create a GetSellerInfo that will use the username of the buyer and the vin of the post they are currently on. This will be sent to the CommunicationEstablishmentManager to search the database for the seller’s username.

- **Communication Establishment (CE-2)(19)**
  1. xUnit tests(3)
     a. Create a test for retrieving chats from the database and will use the username of the user to get all chats associated with them
     b. Create a test that messages are saved in the the datastore and are able to be retrieved

2. Manager Layer(5)
    a. Will use GetChats sent by the entry point that will use IUserAccount to get the necessary information of the user such as the username and sent it to the service layer
    b. Manager layer will be used again with SendMessage that will use the IChat to create an object to be sent to the database that will contain the new message in the chat
3. Service Layer(3)
    a. Create ChatRetrieval service that will return a collection of chat sessions
    b. Another service will be InsertMessageSql that will use the IChat to insert a new message and timestamp of the message into the database
4. Front-End()5
    a. A page will be created to show all the chat sessions of the user and each chat session will be selectable.
    b. Once a chat has been selected it will display the previous messages of the chat session and the user is able to send new messages
5. Entry point(3)
    a. GetMyChats function will be created when the page is visited that will use IUserAccount to get the information necessary of the user in order to retrieve the chat sessions
    b. Then for when a user wants to send a new message PostSentMessage will be created that will use the IChat to create an object to send to the database.

- **Communication Establishment (CE-3)(19)**
    1. xUnit tests(3)
        a. Create a test that when the seller confirms the deal the Marketplace status is changed correctly to Sold
    2. Manager Layer(5)
        a. DealConfirmation will be created that will get the seller's username and the type of deal such as if the deal was canceled or confirmed.
        b. Also create a SendLocationMessage function that will only contain the location of their meetup
        c. For when deal is canceled DealCancellation will be created and will function the same as DealConfirmation but instead the deal type will be different
    3. Service Layer(3)
        a. A service called UpdateDealStatusSql that will change the status of the MarketplaceStatus to 1 by using the VIN or it the deal was canceled it will be change to 0
        b. Will use the InsertMessageSql service again but will only send the location

    4. Front-End(5)
        a. Create a button that confirms the deal
        b. Create a pop up after user confirms the deal that prompts user to enter

location of where to meet up
    c. Create a button that cancels the deal
5. Entry Point(3)
    a. If the user confirms deal it gets the sellers username and creates the deal type in the ConfirmDeal which will be a POST request
    b. If the user cancels the deal similar steps will be taken but the location is never prompted
    c. A post will also be created when the user is prompted to enter the location that will be used to to enter a new message into the chat.

<u>Vehicle Profile</u>

- **Vehicle Profile (VP-1) → Rainier → 5EP**
    1. Entry Point → Rainier → 1EP
        a. Create PostAuthStatusCreateVehicleProfile that will check if a user is authorized to create a vehicle
        b. Create a PostCreateVehicleProfile that will take in vehicle profile data and send it to the VehicleProfileCreationManager to be written to the database. This method will get the UserAccount model from the tokens of the HTTP request before sending the data to the manager.
    2. Front End → Rainier → 4EP
        a. Create a button that will create a popup on the Vehicle Profile View
        b. Create a popup that will let the user choose to input vehicle information manually or use the NHTSA API to gather their vehicle information
        c. Create a form that will be used to let the user input vehicle vin. The vin will be sent to the NHTSA API via Fetch method if they chose so
        d. Show the provided data from the NHTSA API to the user to let them verify that the information from the api is accurate
        e. Create a form that will be used to let the user input VIN, License Plate, Make, Model, Year, Color, and Description of their vehicle. Auto fill in the form with the values from the API if they chose to let the API retrieve their vehicle details and if they confirmed that it was accurate.
        f. Submit button will send the data from the form to the PostVehicleProfile HTTP request from the entry point.
- **Vehicle Profile (VP-2) → Rainier → 4EP**
    1. Entry Point → Rainier → 1EP
        a. Create PostAuthStatusModifyVehicleProfile that will check if a user is authorized to modify a Vehicle Profile
        b. Create a PostUpdateVehicleProfile that will take in the new vehicle profile data and send it to the VehicleProfileModificationManager to be written to the database. This method will get the UserAccount model from the tokens of the HTTP request before sending the data to the manager.
    2. Front End → Rainier → 3EP

        a. Add a new button on a vehicle profile that will let a user update a vehicle
        b. Create a popup that will let the user see their vehicle information and a form for information that they want to change. Make, Model, Year, LicensePlate, Color, and Description are the only fields that should be editable. Two buttons will be added, "Submit" and "Cancel"
        c. Submit button will submit the vehicle profile with the updated values to the PostUpdateVehicleProfile entrypoint and update the existing vehicle profile in the session storage.
        d. Cancel button will return the user back to the vehicle profile view without calling the PostUpdateVehicleprofile HTTP request

- **Vehicle Profile (VP-3)** → **Rainier** → **4EP**
    1. Entry Point → Rainier → 1EP
        a. Create PostAuthStatusCreateVehicleProfile that will check if a user is authorized to delete the requested vehicle
        b. Create PostDeleteVehicleProfile that will take in vehicle profile data and send it to the VehicleProfileCreationManager to be written to the database. This method will get the UserAccount model from the tokens of the HTTP request before sending the data to the manager.
    2. Front End → Rainier → 3EP
        a. Add a new button on a vehicle profile that will let a user delete a vehicle
        b. Create a popup that will let the user confirm or deny that they want to delete their vehicle. Two buttons will be made, "Submit" and "Cancel"
        c. Submit button will submit the selected vehicle profile to the PostDeleteVehicleProfile entrypoint and delete the existing vehicle in the session storage.
        d. Cancel button will return the user back to the vehicle profile view without calling the PostDeleteVehicleProfile request
- **Vehicle Profile (VP-4)** → **Rainier** → **2EP**
    1. Front End → Rainier → 2EP
        a. Link the Vehicle Profile view to the Vehicle Profile button in the navigation bar. There is an implementation of the Vehicle Profile view in javascript currently on the main branch, but it is not linked to the front end code yet.
        b. Fix the input values for the View Vehicle Profile javascript functions
- **Vehicle Profile (VP-5)** → **Rainier** → **2EP**
    1. Front End → Rainier → 2EP
        a. Fix the input values for the View Vehicle Profile Details javascript functions. There is a current implementation of the Vehicle Profile details view in the javascript currently on the main branch

Inventory Management

- **Inventory Management (IM-1) → Rainier → 3EP**
    1. xUnit Tests → Rainier → 3EP
        a. Input data types for user are not valid - fail scenario
        b. Retrieval of vehicle that is not in the database, returns no vehicle - pass scenario
        c. Retrieval of vehicles, no vehicles in database, returns no vehicles - pass scenario
- **Inventory Management (IM-2) → Rainier → 4EP**
    1. xUnit Tests → Rainier → 4EP
        a. Input data types for user or search filters are not valid - fail scenario
        b. Retrieval of vehicle that is not in the database, returns no vehicle - pass scenario
        c. Retrieval of vehicles, no vehicles in database, returns no vehicles - pass scenario
        d. Retrieval of vehicles, no vehicles meet the search filter requirements, returns no vehicle - pass scenario

Car News Center

- **Car News Center (CNC-1)** → Vi → 10EP
    1. Entry point for Get request to get all news articles for any vehicles in the datastore → Vi → 2EP
    2. JS script method to handle extracting and generating view for displaying all the new articles ` → Vi → 6EP
    3. XUnit test: → Vi → 2EP
        a. 1 for fail case when news articles are not received from API → Vi → 1EP
        b. 1 for fail case when vehicles are not retrieved successfully from datastore → Vi → 1EP
- **Car News Center (CNC-3)** → Vi → 11EP
    1. Entry point for Get request to show all current alerts in datastore → Vi → 2EP
    2. Entry point for Post request to create new alerts to datastore → Vi → 2EP
    3. JS Script for extracting and generating view for displaying all notifications in datastore → Vi → 6EP
    4. XUnit test: → Vi → 1EP
        a. 1 for fail case when notifications object are not retrieved from datastore →Vi → 1EP

Vehicle Marketplace

- **Vehicle Marketplace (VPM-1)** → Vi → 7EP
    1. Entry point for Post request to upload vehicle to marketplace → Vi → 2EP
    2. JS Script to generate view when the upload process is success → Vi → 3EP
    3. XUnit test: →Vi → 2EP
        a. 1 for fail case when vehicles is not uploaded successfully to the datastore →Vi → 1EP
        b. 1 for success case when vehicles are uploaded successfully to datastore →Vi → 1EP
- **Vehicle Marketplace (VPM-2)** → Vi → 7EP
    1. Entry point for Post request to delete vehicle from marketplace →Vi → 2EP
    2. JS script to generate view when the delete process is success →Vi → 3EP
    3. XUnit test: →Vi → 2EP
        a. 1 for fail case when vehicle is not deleted successfully from the datastore →Vi → 1EP
        b. 1 for success case when vehicle is successfully deleted from datastore →Vi → 1EP
- **Vehicle Marketplace (VPM-3)** →Vi → 12EP
    1. Entry point for Get request to view all vehicles on marketplace →Vi → 2EP
    2. JS Script to extract and generate view for displaying all the vehicles on the marketplace →Vi → 6EP
    3. XUnit test: →Vi → 4EP
        a. 1 for fail case when vehicles are not fetched successfully from the datastore →Vi → 1EP
        b. 1 success case for pagination when retrieve all vehicles from datastore →Vi → 1EP
        c. 1 fail case for pagination when retrieve all vehicles from datastore →Vi → 1EP
        d. 1 success case for all vehicles fetched successfully from datastore →Vi → 1EP

Scrap Your Car

- **Scrap Your Car Design (SYC-1)** → **Jesus** → **12 EP**
    1. List all Situations that must be tested for (BRD Pass Fail Scenarios)

        - 1 EP
    2. DTO: Car Part
        - 0.5 EP
    3. DTO: Car Part Listing
        - 0.5 EP
    4. Sequence Diagram: Create Listing (Service + Target)
        - 1 EP
    5. Sequence Diagram: Retrieve Part DTO (Service + Target)
        - 0.5 EP
    6. Sequence Diagram: Retrieve Listing (Service + Target)
        - 0.5 EP
    7. Sequence Diagram: Update Listing (Service + Target)
        - 0.5 EP
    8. Sequence Diagram: Delete Listing (Service + Target)
        - 0.5 EP
    9. Sequence Diagram: Create Default Listing(All Car Parts) (Manager + End Point)
        - 0.5 EP
    10. Sequence Diagram: Edit Listing (Manager Layer + End Point)
        - 0.5 EP
    11. Visual Design/Drawing: Default View with buttons (View)
        - 2 EP
    12. Sequence Diagram: “List A Car For Scrap” Front End (View + Script)
        - 2 EP
    13. Sequence Diagram: “See/Edit My Listings” Front End (View + Script)
        - 2 EP
- **Scrap Your Car Design (SYC - 3) → Jesus → 4.5 EP**
    1. List all Situations that must be tested for (BRD Pass Fail Scenarios)
        - 1 EP
    2. Visual Design/Drawing: View with “Create Part Listing” Button (View)
        - 2 EP
    3. Sequence Diagram: Create Individual Part Listing (Manager Layer + Entry Point)
        - 0.5 EP
    4. Sequence Diagram: “Create Part Listing” Front End (View + Script)
        - 1 EP
- **Scrap Your Car Design (SYC - 4) → Jesus → 8 EP**
    1. List all Situations that must be tested for (BRD Pass Fail Scenarios)
        - 1 EP
    2. DTO: Search Parameters
        - 0.5 EP
    3. Visual Design/Drawing: View with Options for Search Parameters (View)

        - 2 EP
    4. Visual Design/Drawing: View with Parts shown, after search (View)
        - 2 EP
    5. Written Design: “GetParts” Function, SQL Design (Target Layer)
        - 1 EP
    6. Sequence Diagram: “GetParts” Function (Target Layer)
        - 0.5 EP
    7. Written Design: “SearchForParts” Function (Service Layer)
        - 1 EP
    8. Sequence Diagram: “SearchForParts” Function (Service Layer)
        - 0.5 EP
- **Scrap Your Car Design (SYC - 5) → Jesus → 10.5 EP**
    1. List all Situations that must be tested for (BRD Pass Fail Scenarios)
        - 1 EP
    2. DTO: Buy Request
        - 0.5 EP
    3. Visual Design/Drawing: View with “My Buy Requests” button (View)
        - 2 EP
    4. Visual Design/Drawing: View with Outgoing Buy Requests (View)
        - 2 EP
    5. Visual Design/Drawing: Search results with “Send Buy Request” button (View)
        - 2 EP
    6. Sequence Diagram: Create Buy Request Function (Service + Target)
        - 0.5 EP
    7. Sequence Diagram: Retrieve Buy Request Function (Service + Target)
        - 0.5 EP
    8. Sequence Diagram: Delete Buy Request Function (Service + Target)
        - 0.5 EP
    9. Sequence Diagram: Send a Request (Manager layer + End Point)
        - 0.5 EP
    10. Sequence Diagram: Get all Requests (Manager layer + End Point)
        - 0.5 EP
    11. Sequence Diagram: Delete Request (Manager Layer + End point)
        - 0.5 EP
- **Scrap Your Car Design (SYC - 6) → Jesus → 5.5 EP**
    1. List all Situations that must be tested for (BRD Pass Fail Scenarios)
        - 1 EP
    2. DTO: Contact Info
        - 0.5 EP
    3. Visual Design/Drawing: Outgoing/Incoming Requests View w/ buttons (View)

    - 2 EP
4. Sequence Diagram: Update Buy Request Function (Service + Target)
    - 0.5 EP
5. Sequence Diagram: Create Contact info (Service + Target)
    - 0.5 EP
    - Note: This will be associated with listing, shared Retrieve function
6. Sequence Diagram: Accept Request (Manager + End Point)
    - 0.5 EP
7. Sequence Diagram: Deny Request (Manager + End Point)
    - 0.5 EP

<u>Logging Web API</u> **→ Jesus** **→ 6.5 EP**

- **Web API**
    1. Implementation: Logging Entry Point + CORS Setup
        - 0.5 EP
    2. Implementation: Logging Controller + End Point
        - 0.5 EP
- **Front End Function**
    1. Implementation: “CreateLog()” Function (Script)
        - 0.5 EP
    2. Testing: “CreateLog()” Front end Testing
        - 5 EP

<u>Service Log</u>

- **Service Log (SL 1 – 4)** **→ Jason** **→ 12.7 EP**
    1. Revise ServiceLogCreation_Pass sequence diagram → Jason → .5
    2. Revise ServiceLogCreation_BusinessRule_Fail diagram → Jason → .5
    3. Revise ServiceLogCreation_DataStore_Fail diagram → Jason → .5
    4. Revise Service Log class diagram Functions → Jason → .5
    5. Add CreateServiceLog Claim to claims list in Excel → Jason → .2
    6. 4/24 meeting Vong for feedback on SL 1- 4 LLD → Jason → .1
    7. Create Service Log Model → Jason → .5
    8. Create Service Log Service Class → Jason → 1
    9. Create Service Log Sql Db Target Class → Jason → 1
    10. Create PassRequirement 1 test case → Jason → 1
    11. Create Valid Service Log passed test case → Jason → 1
    12. Create Invalid Service Log passed test case → Jason → 1
    13. Create Valid SL passed/ 100 SL exist test case → Jason → 1
    14. Create Sql Db Target createServiceLog method → Jason → 2
    15. Create SL Service ServiceLogCreation method → Jason → 2

- **Service Log (SL 5) → Jason → 2.6 EP**
  1. Revise ServiceLogRetrieveDetails_Pass sequence diagram → Jason → .5
  2. Revise ServiceLogRetriever_Pass sequence diagram → Jason → .5
  3. Revise ServiceLogRetriever_DataStore_Fail diagram → Jason → .5
  4. 4/24 meeting with Vong for feedback on SL 5 LLD → Jason → .1
  5. Research SQL give rank implementation for pagination → Jason → 1

<u>**User Administration**</u>

- **UA - 1 → Jason → 8.6 EP**
  1. Revise CreateAccount _Pass sequence diagram → Jason → .5
  2. Revise CreateAccount_Datastore_Fail sequence diagram → Jason → .5
  3. Revise CreateAccount_Business_Fail sequence diagram → Jason → .5
  4. Create UA - 1 class diagram → Jason → 1
  5. 4/24 meeting Vong for feedback on UA - 1 LLD → Jason → .1
  6. Rewrite CreateValidAccount method → Jason → 2
  7. Rewrite GenerateDefaultClaims method → Jason → 2
  8. Create function call to emailing service → Jason → 2
  9. Add missing log functionalities in BRD to service → Jason → 2
  10. Create new User Administration Manager → Jason → 1
  11. Create new User Administration Manager interface → Jason → 1
  12. Check for valid business rule (DOB) in old code → Jason → 1
      - Date of Birth
      - Username
      - Account Type
- **UA - 2 → Jason → 3.6 EP**
  1. Create RecoverAccount success sequence diagram → Jason → .5
  2. Create SendAdminRequest success sequence diagram → Jason → .5
  3. Create ConfirmRequest success sequence diagram → Jason → .5
  4. Create RecoverAccount_Fail sequence diagram → Jason → .5
  5. Create SendAdminRequest_Fail sequence diagram → Jason → .5
  6. Create ConfirmRequest_Fail sequence diagram → Jason → .5
  7. Create UA - 2 Class diagram → Jason → .5
  8. 4/24 meeting Vong for feedback on UA - 2 LLD → Jason → .1

- **UA - 3 → Jason → 2.1 EP**
  1. Create DeleteAccount success sequence diagram → Jason → .5
  2. Create DeleteAccount_Datastore_Fail sequence diagram → Jason → .5
  3. Create DeleteAccount_Business_Fail sequence diagram → Jason → .5
  4. Create UA - 3 Class diagram → Jason → .5
  5. 4/24 meeting Vong for feedback on UA - 3 LLD → Jason → .1

Task Distribution

Members	**Hour // EP remaining**	**Tasks**
Jason	15 - 12.7 - 2.6 - 8.6 - 3.6 - 2.1 = -14.6	● **Service Log SL 1 – 4 (12.7)** ● **Service Log SL 5 (2.6)** ● **UA - 1 (8.6)** ● **UA - 2 (3.6)** ● **UA - 3 (2.1)**
Jesus	20 - 40.5-6.5 = -27	● **SYC (1,3-6) Design (40.5 EP)** ● **Logging Web API (6.5)**
Vi	28-47= -19	● **Car News Center (21EP)** ○ CNC-1 (10EP) ○ CNC-3 (11EP) ● **Vehicle Marketplace (26EP)** ○ **VPM-1 (7EP)** ○ **VPM-2 (7EP)** ○ **VPM-3 (12EP)**
Gio	20-21-20=-21	**Donate Your Car (DYC - 1) (11)** ● Manager Layer(3) ● Service Layer(2) ● Front End(3) ● Entry Point(3) **Donate Your Car (DYC - 2) (10)** ● Manager Layer(3) ● Service Layer(1) ● Front End(3) ● Entry Point(3) **Communication Establishment (CE-1)(20)** ● xUnit(3) ● Manager Layer(5) ● Models(1) ● Service Layer(3) ● Front End(5) ● Entry Point(3)

Rainier	14 - 5 - 4 - 4 - 2 - 2 - 3 - 4= -10	**Vehicle Profile** ● VP-1 (5 EP) ● VP-2 (4 EP) ● VP-3 (4 EP) ● VP-4 (2 EP) ● VP-5 (2 EP) Inventory Management ● IM-1 (3 EP) ● IM-2 (4 EP)

Final Analysis

Do we accept the estimates for:

Example

Jesus	
Jason	
Vi	
Gio	
Rainier	

Service Log

SL - 1 2 3 & 4 → 12.7 EP	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

SL - 5 → 2.6 EP

Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

User Administration

UA - 1 (8.6 EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

UA - 2 (3.6 EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

UA - 3 (2.1 EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes

Rainier	Yes

Donate Your Car

DYC-1 (11EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

DYC-2 (10EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Communication Establishment

CE-1 (20EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Car News Center

CNC-1 (10EP)

Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

CNC-2 (11EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Vehicle Marketplace

VPM-1 (7EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

VPM-2 (7EP)	
Jesus	Yes
Jason	Yes
Vi	Yes

Gio	Yes
Rainier	Yes

VPM-3 (12EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Vehicle Profile

VP-1 (5 EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

VP-2 (4 EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

VP-3 (4 EP)	

Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

VP-4 (2 EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

VP-5 (2 EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Logging Web API

Logging API (12EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Inventory Management

IM-1 (4EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

IM-2 (4EP)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Scrap Your Car

SYC-1 (12)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

SYC-3 (4.5)	
Jesus	Yes
Jason	Yes
Vi	Yes

Gio	Yes
Rainier	Yes

SYC-4 (8)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

SYC-5 (10.5)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

SYC-6 (5.5)	
Jesus	Yes
Jason	Yes
Vi	Yes
Gio	Yes
Rainier	Yes

Are we within our Sprint Capacity?

Work Item	New Estimate
Donate Your Car DYC-1	11
Donate Your Car DYC-2	10
Communication Establishment CE - 1	20
Communication Establishment CE - 2	19
Communication Establishment CE - 2	19
Car News Center CNC - 1	10
Car News Center CNC - 3	11
Vehicle Marketplace VPM - 1	7
Vehicle Marketplace VPM - 2	7
Vehicle Marketplace VPM - 3	12
Service Log SL - 1	12.7
Service Log SL - 5	2.6
User Admin. UA - 1	8.6
User Admin. UA - 2	3.6
User Admin. UA - 3	2.1
Scrap Your Car Peer Review	2
Scrap Your Car SYC - 1	12
Scrap Your Car SYC - 3	4.5
Scrap Your Car SYC - 4	8
Scrap Your Car SYC - 5	10.5
Scrap Your Car SYC - 6	5.5

Vehicle Profile VP - 1	5
Vehicle Profile VP - 2	4
Vehicle Profile VP - 3	4
Vehicle Profile VP - 4	2
Vehicle Profile VP - 5	2
Inventory Management IM - 1	3
Inventory Management IM - 2	4
Total	222.1

Will We Add More Work

No, we will not add more work due to being over capacity.