

**Core Functionality:** A System for selling Travel Insurance.

**Team Members:**

1. **Lampros Karseras: Policy and Quotes**
2. **Georgios Karseras: Claims**
3. **Joshua Harris: Policy Holder details**
4. **Aman Gulati: Feedback & Recorded Complaints**
5. **Jeff Johnson: Staff details**

**System Overview:**  
This website will allow customers to buy and manage their travel insurance policies.

**System Specification:**  
New customers can generate a selection of travel insurance quotes with varying levels and types of cover on the website based upon how many trips they wish to be covered for, where they’re going, how long for, who with and what they’re doing while they’re away. After the submission of all these details, a Quote is produced based on our Policy backend. Once a customer is happy with their Quote, they can proceed to payment and become a Policy Holder.  
A Policy Holder can access their policy via the same website and make changes where appropriate. There will be a functionality, within their account, which will allow them to view, cancel or renew their existing policy.  
Personal details such as Name, address or contact details can easily be changed however other changes to the policy i.e changes that relate to the sections of insurance may incur an increase in premium or require the policy to be cancelled and reissued to the Policy Holder. The website will also keep a log of the changes made to a policy, when they happened and who made the changes.  
A Policy Holder will also be able to submit Claims via our website. Once all relevant documentation has been received by Tesla Travel Insurance’s claims adjusters, they will be able to decide the outcome of a Policy Holder’s claim. Where a claim is successful, money will be paid directly into the Policy Holder’s bank account.   
Users can also submit feedback or complaints regarding their experience with our services. When complaints are received, appropriate actions must be taken by Tesla Travel Insurance’s customer service agents and ensure that the user who submitted the complaint is happy with how their issues have been dealt with.

# Lampros Karseras Section:

**Allocated Component:**  Policy System

**Developer:**  Lampros Karseras, P2424629

**Overview:**

When a user visits the website, they will be presented with an option to get a travel insurance. The system will force the user to login to their account before getting a policy. If they are not logged in, the system will redirect to the login/register page. The registered user then will be presented with the Policy Front-End, which it will ask for the details of the travel insurance they want to buy. The policy system then it will calculate the price of the insurance based on the selected options from the user. If the user decides to buy the insurance, it will then generate an insurance policy for the specific user, save it to the database and linked it to the user’s account, where they can see it, download it, cancel it or renew it.

The Policy System will also be controlled by the Staff Management System. The staff users will have the ability to add, edit, list, update, find and delete any policy.

Supported functions:

**Add**: Adding a policy, either from the Customer System, or from the Staff System

**Edit**: Staff will have the ability to edit already existing policies on the policy database.

**Update**: The users will have the ability to extend their policy only. They won’t be able to do any other edit. The Staff users, as mentioned above, will have the ability to completely edit the policies.

**Delete**: The users will have the ability to cancel (delete) their own policy. The Staff users will be able to delete any policy in the database.

**List**: The Staff users will be able to list all the policies from the database

**Find/Filter**: The ability for the Staff users to find/filter a specific policy or policies that match a given predicate. The users will be able to see their own policy.

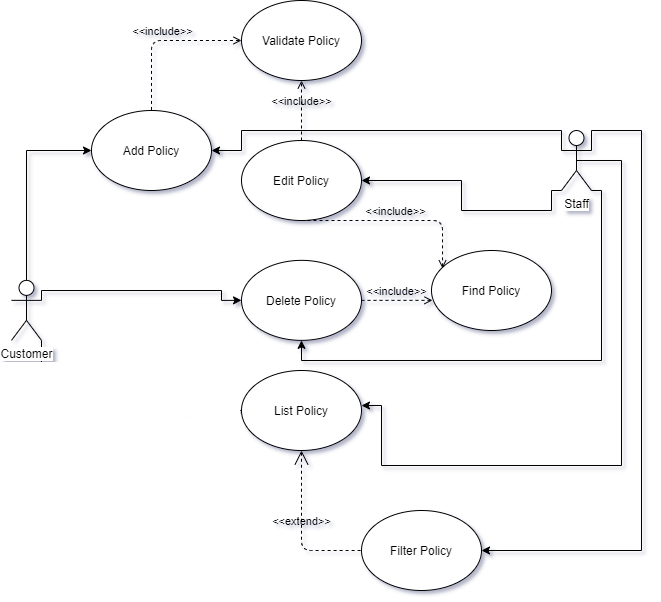
**Validation**: All the incoming commands/details/options will be validated and verify that are in correct, and expected syntax and type.

**Table Schema:**

**Title**: tblPolicy

|  |  |  |  |
| --- | --- | --- | --- |
| PolicyID | Int | Not Null | Primary Key |
| StaffID | Int | Not Null | Foreign Key |
| CustomerID | Int | Not Null | Foreign Key |
| StartDate | Date | Not Null |  |
| Accepted | Boolean | Not Null |  |
| PolicyDetails | Text | Null |  |
| Price | Decimal(8,2) | Not Null |  |

**Use Case Diagram:**



**Use Case Descriptions:**

|  |  |
| --- | --- |
| Use Case Name (Short two- or three-word name) | Add Policy |
| Use Case Description (Short description) | To Add a Policy to the database. Either the customer creates it, or a staff member creates it for a customer. |
| Use Case Author(s) (Who wrote this) | Lampros Karseras |
| Actor(s) (Who does this) | Customer/Staff |
| Locations (Where does this happen) | Back-End functionality |
| Primary pathway (What is the normal “happy path” for this use case?) | Policy is added to database.   1. Customer/Staff click the “Add” button. 2. Customer/Staff redirected to Add page. 3. Customer/Staff make valid choices. 4. Customer/Staff click the “Ok” button. 5. The new Policy is added to database. |
| Alternate pathways (What other paths are there that are not the “happy path”?) | 4a) Customer/Staff made an invalid choice on the Add page.   * + 1. A message is displayed describing the invalid choice and what was the expected values. |
| Exception pathways (What could possibly go wrong?) | Database connection error.  A message is displayed asking to repeat the process later. |

|  |  |
| --- | --- |
| Use Case Name (Short two- or three-word name) | Edit Policy |
| Use Case Description (Short description) | To Edit a Policy already in the database. |
| Use Case Author(s) (Who wrote this) | Lampros Karseras |
| Actor(s) (Who does this) | Staff |
| Locations (Where does this happen) | Back-End functionality |
| Primary pathway (What is the normal “happy path” for this use case?) | A Policy is changed and saved in the Database.   1. Selects a specific policy. 2. Makes changes to the Policy’s attributes. 3. Clicks “Ok” button. 4. Returns a success message. |
| Alternate pathways (What other paths are there that are not the “happy path”?) | 3a) Staff makes invalid changes to the Policy’s attributes   1. System displays error message for each invalid choice |
| Exception pathways (What could possibly go wrong?) | Database connection error.  A message is displayed asking to repeat the process later. |

|  |  |
| --- | --- |
| Use Case Name (Short two- or three-word name) | Delete Policy |
| Use Case Description (Short description) | To Delete a Policy already in the database. |
| Use Case Author(s) (Who wrote this) | Lampros Karseras |
| Actor(s) (Who does this) | Customer/Staff |
| Locations (Where does this happen) | Back-End functionality |
| Primary pathway (What is the normal “happy path” for this use case?) | A Policy is removed from Database.   1. Customer selects its own policy. 2. Clicks “Delete” button. 3. System verifies the action. 4. System removes the Policy from Database. 5. Returns success message. 6. Staff filters the list of Policies for a specific PolicyID/CustomerID. 7. Selects a specific policy. 8. Clicks “Delete” button. 9. System verifies the action. 10. System removes the Policy from Database. 11. Returns success message. |
| Alternate pathways (What other paths are there that are not the “happy path”?) | 1a) Staff enters a non-existent PolicyID/CustomerID   1. Returns an empty list |
| Exception pathways (What could possibly go wrong?) | Database connection error.  A message is displayed asking to repeat the process later. |

|  |  |
| --- | --- |
| Use Case Name (Short two- or three-word name) | List Policy |
| Use Case Description (Short description) | To List Policies already in the database. |
| Use Case Author(s) (Who wrote this) | Lampros Karseras |
| Actor(s) (Who does this) | Staff |
| Locations (Where does this happen) | Back-End functionality |
| Primary pathway (What is the normal “happy path” for this use case?) | List Policies based on PolicyID/CustomerID   1. Customer/Staff member enters the system. 2. A list of all Policies in the Database is displayed. |
| Alternate pathways (What other paths are there that are not the “happy path”?) | 2a) There aren’t any data in the system.   1. A message is displayed for empty list. |
| Exception pathways (What could possibly go wrong?) | Database connection error.  A message is displayed asking to repeat the process later. |

|  |  |
| --- | --- |
| Use Case Name (Short two- or three-word name) | Validate Policy |
| Use Case Description (Short description) | To Validate Policy Attributes |
| Use Case Author(s) (Who wrote this) | Lampros Karseras |
| Actor(s) (Who does this) | Customer/Staff |
| Locations (Where does this happen) | Back-End functionality |
| Primary pathway (What is the normal “happy path” for this use case?) | Validate the Policies Attributes   1. Customer/Staff creates or make changes to a Policy. 2. System returns success message. |
| Alternate pathways (What other paths are there that are not the “happy path”?) | 1a) Customer/Staff makes invalid choices or changes.   1. A message displaying the incorrect choice/change and what is expected. |
| Exception pathways (What could possibly go wrong?) | System error.  A message is displayed asking to repeat the process later. |

|  |  |
| --- | --- |
| Use Case Name (Short two- or three-word name) | Find Policy |
| Use Case Description (Short description) | To Find a specific Policy |
| Use Case Author(s) (Who wrote this) | Lampros Karseras |
| Actor(s) (Who does this) | Customer/Staff |
| Locations (Where does this happen) | Back-End functionality |
| Primary pathway (What is the normal “happy path” for this use case?) | Find a specific Policy   1. Customer/Staff selects a specific Policy. 2. System returns all the attributes of that Policy. |
| Alternate pathways (What other paths are there that are not the “happy path”?) | 1a) Staff search for non-existent Policy   1. A “not-found” error message is displayed. |
| Exception pathways (What could possibly go wrong?) | System error.  A message is displayed asking to repeat the process later. |

|  |  |
| --- | --- |
| Use Case Name (Short two- or three-word name) | Filter Policy |
| Use Case Description (Short description) | To Filter a list of Policies |
| Use Case Author(s) (Who wrote this) | Lampros Karseras |
| Actor(s) (Who does this) | Staff |
| Locations (Where does this happen) | Back-End functionality |
| Primary pathway (What is the normal “happy path” for this use case?) | Filters a list of Policies based on CustomerID.   1. Staff list all the Policies in the Database 2. Applies a Filter (CustomerID) 3. The filtered list is displayed. |
| Alternate pathways (What other paths are there that are not the “happy path”?) | 1a) No data in the Database   1. An empty list is returned. A message is displayed.   2a) Staff applies a filter for non-existent CustomerID.   1. An empty list is returned. |
| Exception pathways (What could possibly go wrong?) | System error.  A message is displayed asking to repeat the process later. |

# **Georgios Karseras Section:**

**Allocated Component:**  Claims System

**Developer:**  Georgios Karseras, P2424630

**Overview:**

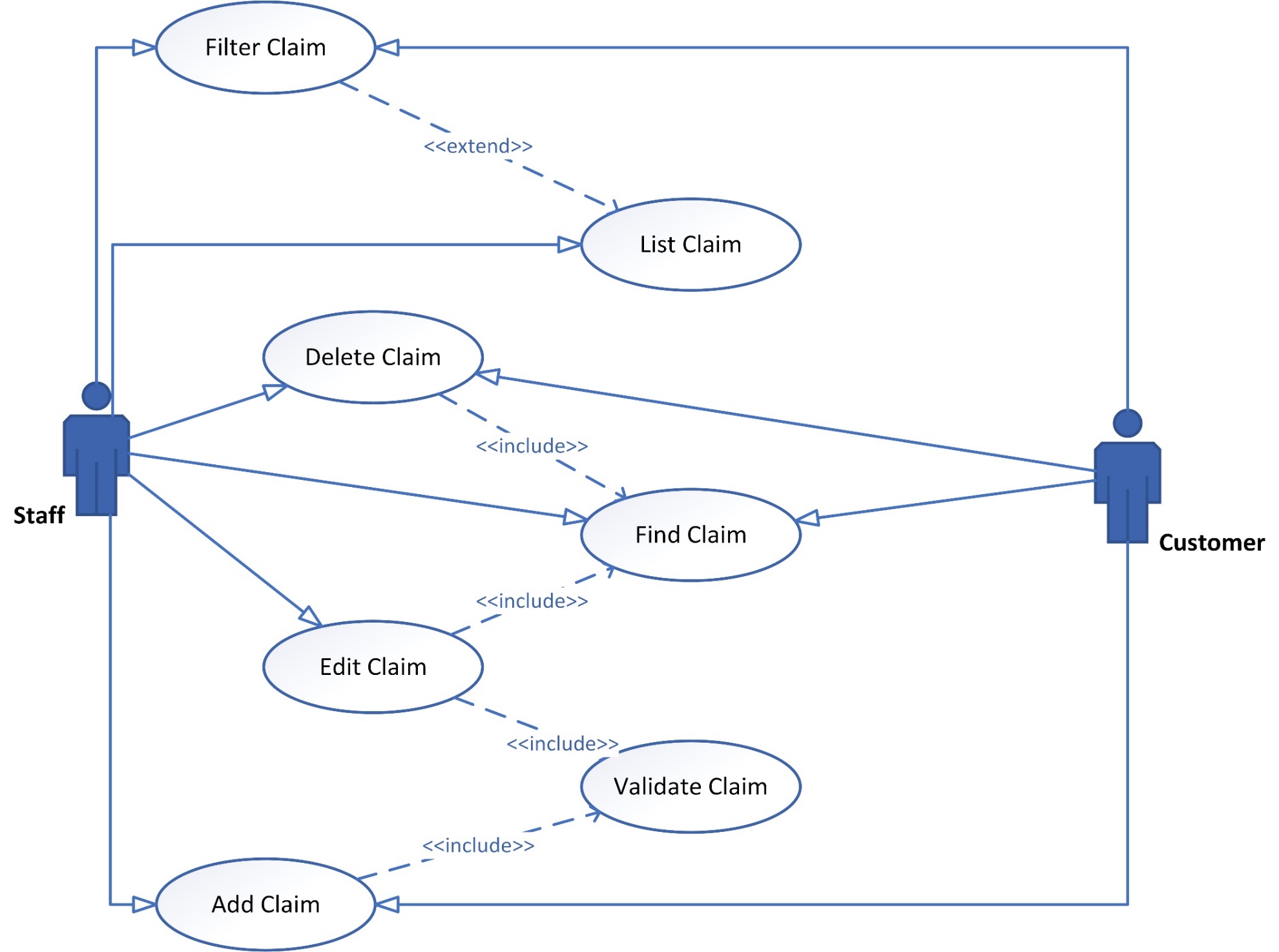
A Policy Holder can use the Claims system front end to make a Claims request. A form with their personal details will be presented alongside pre-defined reasons to select for making this Claim, e.g. cancelled trip, personal incident, hospital bills etc. After they complete the form and provide the relevant documents, a new Claim entry in the back end will be created. Staff will then have to look at the Claim entries, to verify their validity in accordance to the provided documents, to either approve or decline the Claim and finally to calculate the return amount in case it is approved.

**Add –** Customer or Staff Adds a new Claim entry.  
**Edit –** Staff can edit Claims from the Database.  
**Update –** Staff can Update Claims from the Database, this will mostly be done on new entries to verify their validity, accept or decline the claim and update the returned amount when necessary.  
**Delete –** Customers can Delete their Claim requests and Staff can Delete a Claim entry from the Database when necessary.  
**List –** Staff can List all the Claims from the Database.  
**Find –** The Customer and Staff can Find specific details on a Claim using the Primary Key, ClaimID.  
**Filter –** Customer can Filter based on their CustomerID to display all of their Claims and Staff can Filter based on a search pattern from the List.  
**Validate –** When editing a pre-existing Claim or when adding a new one, Validation is done on all fields so that they match the data types specified in the Database and other constrictions that are maybe required(for example you cannot create a Claim for a future Date).

**Individual Table Schema:**

**Title: tblClaim**

|  |  |  |  |
| --- | --- | --- | --- |
| ClaimID | Int | Primary Key | NOT NULL |
| CustomerID | Int | Foreign Key | NOT NULL |
| StaffID | Int | Foreign Key | NOT NULL |
| ClaimReason | Text |  | NOT NULL |
| ClaimDate | Date |  | NOT NULL |
| ClaimAmnt | Decimal |  | NOT NULL |
| ClaimStatus | Boolean |  | NOT NULL |

**USE CASE DIAGRAM:** 

**USE CASE DESCRIPTIONS:**

|  |  |  |
| --- | --- | --- |
| **Use Case:** | Filter Claim. | |
| **Case Description:** | Filter from a List of Claims. | |
| **Actor(s):** | Customer & Staff. | |
| **Locations:** | Front End. | |
| **Primary pathway:** | 1. User enters the List web site. | 1. System displays all Claims in Database. |
| 1. User Filters the List with a Claim Reason. | 1. The new Filtered List is being displayed. |
| **Alternate pathways:** | The Claim Reason given doesn’t exist in the list.   1. System will display an empty List.   Server or Database error.   1. System displays an appropriate error message and asks the User to try again later. | |

|  |  |  |
| --- | --- | --- |
| **Use Case:** | List Claim. | |
| **Case Description:** | List all the Claims in the Database. | |
| **Actor(s):** | Staff. | |
| **Locations:** | Back End. | |
| **Primary pathway:** | 1. User enters the List web site. | 1. System displays all Claims in Database. |
| **Alternate pathways:** | Server or Database error.   1. System displays an appropriate error message and asks the User to try again later. | |

|  |  |  |
| --- | --- | --- |
| **Use Case:** | Delete Claim. | |
| **Case Description:** | Delete a Claim from the Database. | |
| **Actor(s):** | Customer & Staff. | |
| **Locations:** | Back End. | |
| **Primary pathway:** | 1. User enters the List web site. | 1. Customer selects his own Claim request. Staff select any Claim from the Database. |
| 1. Customer/Staff press the Delete button. | 1. The Delete web site is being displayed asking for verification of the action. |
| 1. Customer/Staff can either press NO or YES to proceed in deleting the Claim. | 1. System redirects to the List web site. |
| **Alternate pathways:** | Server or Database error.   1. System displays an appropriate error message and asks the User to try again later. | |

|  |  |  |
| --- | --- | --- |
| **Use Case:** | Find Claim. | |
| **Case Description:** | Find a specific Claim from the Database. | |
| **Actor(s):** | Customer & Staff. | |
| **Locations:** | Back End. | |
| **Primary pathway:** | 1. User enters the Add & Edit web page. | 1. User enters ClaimID to Find from the Database. |
| 1. User press the Find button. | 1. Database returns the Fields that match that ClaimID and Displays the values in the web form. |
| **Alternate pathways:** | ClaimID not found in the Database.   1. The System Displays an appropriate error message.   Server or Database error.   1. System displays an appropriate error message and asks the User to try again later. | |

|  |  |  |
| --- | --- | --- |
| **Use Case:** | Edit Claim. | |
| **Case Description:** | Edit a Claim from the Database. | |
| **Actor(s):** | Staff. | |
| **Locations:** | Back End. | |
| **Primary pathway:** | 1. User enters the List web site. | 1. User selects from the List a Claim to edit. |
| 1. User press Edit button which redirects to the Add & Edit web page | 1. User changes the values and press OK. |
| 1. Database Updates values and redirects user to the List web site. | |
| **Alternate pathways:** | Data given do not match the Database Data types.   1. The System Displays an appropriate error message.   Date given is in the future or over 2 months in the past.   1. The System Displays an appropriate error message.   Server or Database error.   1. System displays an appropriate error message and asks the User to try again later. | |

|  |  |  |
| --- | --- | --- |
| **Use Case:** | Validate Claim. | |
| **Case Description:** | Validate a new Claim add or edit Fields. | |
| **Actor(s):** | Customer & Staff. | |
| **Locations:** | Back End. | |
| **Primary pathway:** | 1. User enters the Add & Edit web site. | 1. User either creates a new Claim or makes changes to a pre-existing one. |
| 1. User press the OK button | 1. System validates data entered. |
| **Alternate pathways:** | Data given do not match the Database Data types.   1. The System Displays an appropriate error message.   Date given is in the future or over 2 months in the past.   1. The System Displays an appropriate error message.   Server or Database error.   1. System displays an appropriate error message and asks the User to try again later. | |

|  |  |  |
| --- | --- | --- |
| **Use Case:** | Add Claim. | |
| **Case Description:** | Add a new Claim to the Database. | |
| **Actor(s):** | Customer & Staff. | |
| **Locations:** | Back end. | |
| **Primary pathway:** | 1. User enters the Add & Edit web site. | 1. System displays Form necessary to fill to create a new Claim. |
| 1. User fills the fields and presses the button OK. | 1. System Adds the new Claim to the Database. |
| **Alternate pathways:** | Data given do not match the Database Data types.   1. The System Displays an appropriate error message.   Date given is in the future or over 2 months in the past.   1. The System Displays an appropriate error message.   Server or Database error.   1. System displays an appropriate error message and asks the User to try again later. | |

# Jeff Johnson Section:

**Individual Spec**

Within the system we have decided to create, I have been entrusted to deal with the staff component, and in this component, there are many functionalities that I must follow e.g.:

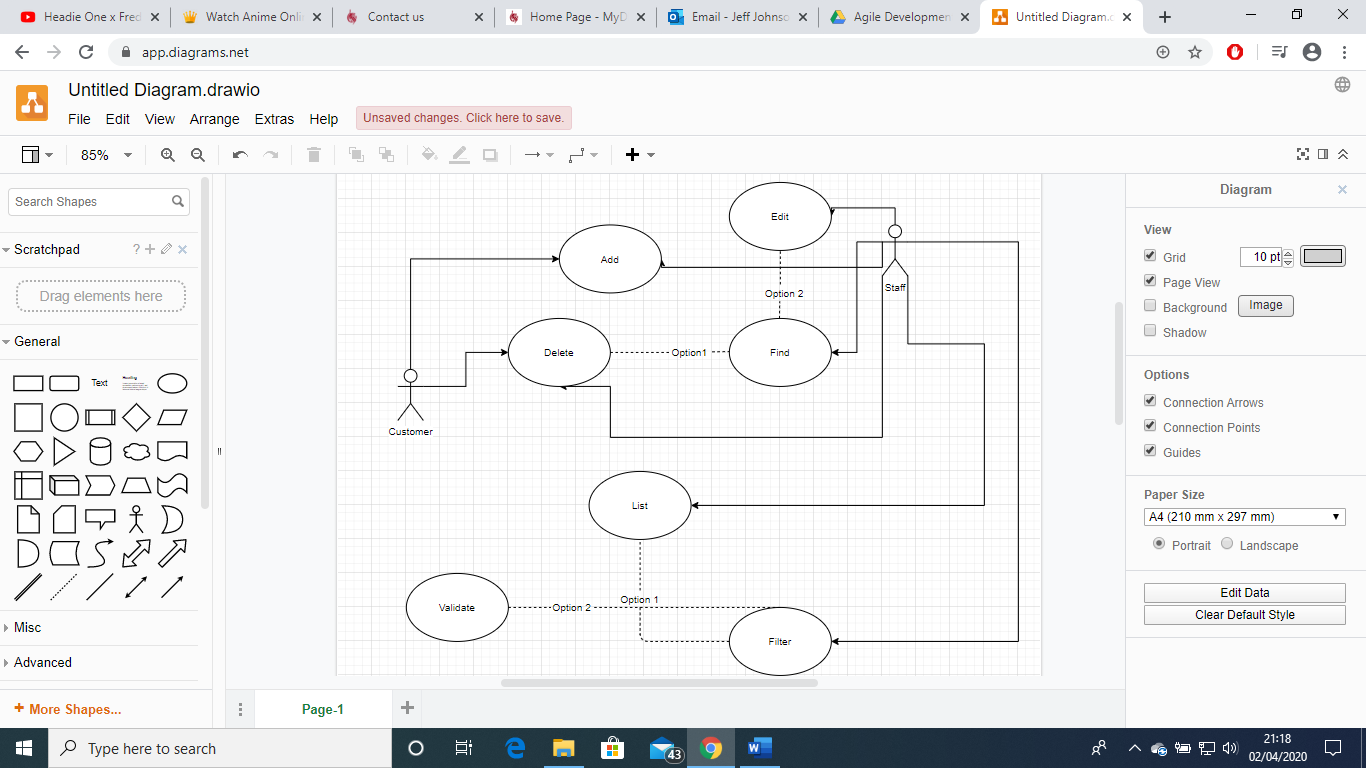
* Add
* Edit
* Delete
* List
* Find
* Filter
* Validate

But since our system is based on travel insurance, the staff will have the responsibility of giving good customer service. If customers have any queries they should be able to call and our staff will deliver any possible assistance. During the development phase I hope to use my knowledge in SQL to retrieve any data from the text box so that I can concatenate the data from the text box to the database. By doing this, I can then add entries to the database if it’s a new customer, whereas if they’re an existing customer I can then edit their details if there are new information for that customer and I can delete them if they wish to be not part of our company. But as a developer I should be making the code easier for myself and colleagues by having the option to list/filter all customers so that the job of finding a customer is easy. And finally, I would need to have the option of validating the customer.

**Schema Table**

|  |  |
| --- | --- |
| Staff\_ID | Primary key/Integer |
| Name | String |
| Address | String |
| Salary | Real |
| ContactNo | Integer |
| IsValid | Boolean |

**Use Case Diagram:**



**Use Case Descriptions:**

|  |  |
| --- | --- |
| Use Case | Get Staff Details |
| Description | Manager or team leader can view staff details |
| Author | Jeff Johnson |
| Actor(s) | Developers and employees |
| Primary pathway | Get Staff Details  Project leader selects staff details component  Staff details are then accessible by project leader |
| Alternative pathway | Some details might be incomplete  The user will then be redirected to update the incomplete form |
| Exception pathway | Error |

|  |  |
| --- | --- |
| Use Case | Add Staff Details |
| Description | Manager or team leader can add staff details |
| Author | Jeff Johnson |
| Actor(s) | Developers and employees |
| Primary pathway | Project leader selects staff details component  Add entry - Staff Details  Staff details is then updated by project leader |
| Alternative pathway | Incomplete forms will be stored so when database is next accessed data is not lost  The user will then be able to update the incomplete form |
| Exception pathway | Error |

|  |  |
| --- | --- |
| Use Case | Edit Staff Details |
| Description | Manager or team leader can edit staff details |
| Author | Jeff Johnson |
| Actor(s) | Developers and employees |
| Primary pathway | Project leader selects staff details component  Edit Staff Details  Staff details are then updated by project leader |
| Alternative pathway | Project leader should also have the option to access and edit multiple tables |
| Exception pathway | Error |

|  |  |
| --- | --- |
| Use Case | Delete Staff Details |
| Description | Manager or team leader can delete staff details |
| Author | Jeff Johnson |
| Actor(s) | Developers and employees |
| Primary pathway | Project leader selects staff details component  Delete the selected staff details  Staff details (table) is then updated |
| Alternative pathway | If user cancels system should redirect users to the home page |
| Exception pathway | Error |

|  |  |
| --- | --- |
| Use Case | Show Staff Details |
| Description | Manager or team leader can display staff details |
| Author | Jeff Johnson |
| Actor(s) | Developers and employees |
| Primary pathway | Project leader selects staff details component  List the following table - Staff Details  All staff details are then displayed |
| Alternative pathway | Project leader should have the availability of filtering the shown table |
| Exception pathway | Error |

|  |  |
| --- | --- |
| Use Case | Filter Staff Details |
| Description | Manager or team leader can filter staff details |
| Author | Jeff Johnson |
| Actor(s) | Developers and employees |
| Primary pathway | Project leader selects staff details component  Filter Staff Details  All staff details that meet the filtered conditions are then displayed |
| Alternative pathway | Within the filtering component the project leader should be able to validate anyone. |
| Exception pathway | Error |

|  |  |
| --- | --- |
| Use Case | Validate Staff Details |
| Description | Manager or team leader can validate staff details |
| Author | Jeff Johnson |
| Actor(s) | Developers and employees |
| Primary pathway | Project leader selects staff details component  Validate Staff Details  Staff details (table) is then updated by project leader |
| Alternative pathway | If user cancels they will be redirected to home page |
| Exception pathway | Error |

# **Joshua Harris Section:**

**Individual Overview:**

A new customer can use our website to generate travel insurance policies with varying levels of cover suitable for their travel needs. Should a traveller need to claim on the policy, they can use our website to check the sections of insurance and submit their claim, plus any documentation that may be required by the claims adjustors. Once a claim is successful, money is paid directly into the policy holder’s bank account.

I will build the component which stores the personal details of the Policy Holder. These details and their data types are:

|  |  |
| --- | --- |
| **Variable Name** | **Data Type** |
| Customer ID | String |
| Name | String |
| Date of Birth | Date |
| Address | String |
| Postcode | String |
| E-mail address | String |
| Contact Number | Numeric String |
| Marketing Preferences | Boolean |

When a user generates a quote, the system will collect the customer’s personal details. This is so a customer’s identity can be verified upon the completion of some data protection questions whenever they get into contact with a staff member so they can access their Policy; they are required to submit personal details that are the same as what is shown on their travel documents to ensure their insurance isn’t invalidated. Contact details such as phone, email and postal address will also be collected so the customer can be contacted with regards to their insurance policy by staff members. Marketing Preferences can also be set by the customer where they may wish to receive additional news and info related to their travel insurance policy.

In order for these relevant customer details to be collected, a customer must complete and submit a web form which will set each field within my ‘customer details’ component to the user’s input, which will be validated, sanitised and filtered to ensure the application remains secure and only appropriate user input can be accepted.

Once all the relevant fields have been set, a new customer can be added to a collection of customers with a unique, randomly generated ID number stored within the primary key field: ‘*CustomerID*’. This is so when a user provides *CustomerID*, they can find that customer in the customer collection, otherwise they will need to use the customer’s other personal details until the user finds what they are looking for.

If a customer wishes for their personal details to be deleted and they do not have either an insurance policy or a claim being processed, then they can request to be deleted from the system.

**Use Case Descriptions:**

|  |  |
| --- | --- |
| Use Case | Get Customer’s Details |
| Description | The user views the Customer’s details. |
| Use Case Author | Joshua Harris |
| Actor(s) | Customers, Staff Members and System admins. |
| Locations | Front and back end of system. |
| Primary pathway | Get Customer Details  The user selects the Customer details component.  A given customer’s details are displayed to the user. |
| Alternate pathways | Some of the details relating to the Customer are incomplete or inaccurate.  Customer amends relevant fields accordingly and saves changes.  Customer Details are now accurate. Changes have been logged by the system. |
| Exception pathways | Database Connection fails.  Error message tells the user what’s gone wrong. |

|  |  |
| --- | --- |
| Use Case | Set Customer’s Details |
| Description | The user changes the Customer’s details. |
| Use Case Author | Joshua Harris |
| Actor(s) | Customers, Staff Members and System admins. |
| Locations | Front and back end of system. |
| Primary pathway | Set Customer Details  The user selects the Customer details component and proceeds to amend fields which they can see are incomplete or inaccurate.  Once the user is happy that the Customer’s details are now accurate, changes are saved and logged by the system. |
| Alternate pathways | The user sees the details relating to the Customer are already accurate.  No need to make any amendments or log any changes to Customer details. |
| Exception pathways | Database Connection fails.  Error message tells the user what’s gone wrong. |

|  |  |
| --- | --- |
| Use Case | Delete Customer’s Details |
| Description | The user deletes the Customer and their details. |
| Use Case Author | Joshua Harris |
| Actor(s) | Customers, Staff Members and System admins. |
| Locations | Front and back end of system. |
| Primary pathway | Delete Customer Details  The user selects the Customer details component and decides to delete the customer and their details from the system.  The system is checks if the customer currently has an active policy or a claim that’s being processed so only Customer’s who do not hold a travel insurance policy can be deleted. If so, the user is prompted with the confirmation message “Are you sure you wish to delete the customer’s details?”. When the user clicks yes, the customer’s details are deleted and the customer is removed from the customer collection. |
| Alternate pathways | The Customer still has an active insurance policy or claim being processed and so cannot delete their details until their policy ends and any claims are resolved. |
| Exception pathways | Database Connection fails.  Error message tells the user what’s gone wrong. |

|  |  |
| --- | --- |
| Use Case | Add Customer to Customer Collection |
| Description | The user submits the details for a new Customer which now needs to be added to the Customer Collection. |
| Use Case Author | Joshua Harris |
| Actor(s) | Customers, Staff Members and System admins. |
| Locations | Front and back end of system. |

**Entity Relationship Diagram (ERD) for Tesla Travel Insurance:**

**A close up of a map

Description automatically generated**

**Class Diagram:**A screenshot of a cell phone

Description automatically generated