## 

## XYZ Driver Association Case Study

Table of Contents

[XYZ Driver Association Case Study 1](#_Toc19724971)

[Specifications 1](#_Toc19724972)

[Test cases 2](#_Toc19724973)

[Marking Process 2](#_Toc19724974)

[Sprint 1 3](#_Toc19724975)

[Sprint 2 3](#_Toc19724976)

[Sprint 3 3](#_Toc19724977)

[Important Notes: 4](#_Toc19724978)

### Specifications

XYZ Drivers Association decides to set up a solidarity fund to subsidise members for minor accidents so that the members can avoid making claims to Insurance companies. The association provides a certain amount of subsidy once a claim is made, and sums up all claims made to the end of each year. The members are expected to pay annual membership fee and the allocated portion of the total annual charge from lumpsum of the claims. The charges are allocated annually. Members will be able to make claims after 6 months of their membership, be allowed to make maximum 2 claims per year, and will not be supported if these circumstances are not met.

The Association aims to develop a web application (**WebApp**) to streamline its business processes and help for a smooth administration. The **WebApp** will let a member log on (register first if required), pay fees and outstanding balances, and be able to make a claim. Registrations for membership require personal information including name, address, date of birth, and date of registration of the applicants. (A **web service** may/should be used for address lookup purposes). Once this information is submitted, the system should record it and confirm a provisional membership returning a user name and automatically generated password as well as charging him/her with annual membership fee. Once the membership fee is paid and been confirmed by the office (admin), the person is upgraded to full membership level. Members are suspended if they did not pay annual fee and charges, and will be allowed to resume their membership once they cleared outstanding balances.

An administrator is expected to process outstanding operations raised by the full members including confirmation of the payments, charge the members based on annual lumpsum and membership fee, and assess the eligibility of the members for the outstanding claims. They require having some facilities to browse through the list of members, search for members based on provided particular information.

The Web Application is expected to meet the following requirements:

1. At least the following pages are expected to be included:
   1. The main (home) page letting users select the type of user and action ahead
   2. Login/registration page for member users is required
   3. A Dashboard page for member users
   4. A management Dashboard page for admin users that lets to process the operations as required
2. A user should be able to navigate through the pages, smoothly, and especially be able to access to its own dashboard and the home page from any page.
3. A member should remain logged on until either the session is timeout or user is changed. With this respect, access to the members’ dashboard should be secured.
4. A web service should be developed for a particular business process ***( e. g. the process for assessing if the members are eligible for support once they make claims)***, deployed and made up-running on a server (e.g. GlassFish) as a separate project, and then be invoked/integrated in the system as a service.
5. The groups are expected to use GitHub as the version control software to ease/facilitate collaboration within their groups and to create evidence of their team work.
6. The whole system should be using
7. Java EE components following MVC patterns,
8. interacting with database (Java DB), and
9. deployed on a server, which has a container (GlassFish).

### Test cases

During the development you will be supplied with a sequence of test cases, which will be selected from the following functions:

1. Login as typical member user (e.g. user = “member1”, password = ”member1”) and create a session lasting for 20 mins. If not a user yet, register as an applicant for membership
2. See member’s dashboard, which consists of :-
   1. Check for outstanding balance
   2. Make a payment
   3. Submit a claim
   4. List all claims and payments to date
3. Navigate back-to main page to let change user (e.g. change to admin)
4. See Admin’s (management) Dashboard consisting of:-
   1. List all members
   2. List all outstanding balances
   3. List all claims
   4. List all provisional member applications
   5. Process individual claims
   6. Process membership applications and upgrade if payment is made
   7. Suspend/Resume membership
   8. Report annual turnover including total income and total pay-outs

Additionally - it is expected that the system uses "**Filtering**" for authentication purposes, especially for **authenticating admin user** - a clear example of this would also be favourably considered (“**Filtering**” is discussed in Chapter 13 of HF book).

You may assume the system will accept cookies. No threading/concurrency considerations are required. All data must be stored and retrieved from Java DB database.

You will be supplied with one SQL script to build tables containing some samples of members, claims and payments. (**XYZ\_Assoc.sql**). This can be customised or extended for a better/optimised DB.

## Marking Process

As a team, you will be asked to demonstrate your developed software system in scheduled practical classes as applicable for each sprint where appropriate. For Sprint 3:

* You must download (from Blackboard) and unzip your project.
* You must run the SQL scripts provided to create and populate the required tables.

Your NetBeans project and your DB server must be runnable on the standard FET configured laboratory machines (or the same configuration on your own laptop which you bring).

* It is your responsibility to attend scheduled classes – failure to demonstrate your system in class will be treated as a non-submission.
* All group members will be awarded the same mark as long as all group members agree that all contributed equally. Otherwise, your individual marks will be generated based on your individual contributions identified based on collected and provided evidence. Everybody must confidentially fill and submit “Peer Assessment Form” for collecting inputs to determine individual contributions. A guiding scheme is included in the document called “Marking Matrix”
* Any group member failing to take part in the demonstration will be assessed as a non-submission and given zero marks.

The quality of your verbal expression in this demonstration is important – incoherent explanations will not achieve high marks. Please be advised that demonstrations last for a fixed-durations, so be prepared to concisely demonstrate and explain your system.

**Sprint-wise expectations** are as follows: -

### Sprint 1

A 10-minute presentation + Q/A session is needed from each group while all group members must attend. A set of slides can be used for presentation alongside **(1)** detailed plan, ideally in the form of Gantt Chart, and **(2)** a Task Delivery Form is needed to be filled, signed and submitted to the tutor, and **(3)** a quick demonstration of Login function.

Sprint 1 carries a weighting of 10% of the overall coursework marks.

Please see the 1st matrix in “Marking Matrix” document for Sprint 1.

### Sprint 2

Completion of backend architecture and partial completion of system functionality including some servelets and front-end user interface. A 10-minute presentation + Q/A session is needed from each group while all group members must attend. A set of slides can be used for presentation; **(1)** detailed plan for the remaining tasks, ideally in the form of Gantt Chart, **(2)** a Task Delivery Form is needed to be filled, signed and submitted to the tutor, and **(3)** a quick demo of few completed functionalities.

Sprint 2 carries a weighting of 30% of the overall coursework marks.

Please see the 2nd matrix in “Marking Matrix” document for Sprint 2.

### Sprint 3

Sprint 3 is the final stage of the work, requires completing all requirements. It carries a weighting of 60% of the overall coursework marks.

Please see the 3rd matrix in “Marking Matrix” document for Sprint 3.

For sprint 3, additionally –

1. A clear and understandable use of "**Web Services**" is requested that be incorporated into the system - merely for demonstration purposes. This can be for delivering one of the main functionalities required.
2. The members’ addresses can be looked up via **Google map services** or a similar one*.*
3. It can also be useful for the system to use "**Filtering**" for admin user authentication purposes- a clear example of this would also be favourably considered (“**Filtering**” is discussed in Chapter 13 of reference book, HF Servlet & JSP).

## Important Notes:

Please also be advised that:

1. Teams are expected to use GitHub as the version control software to ease collaboration within the groups and to create evidence of their team work. Other software development communication tools are available and may also be used to generate evidence of team work.
2. ***A Task Delivery Form*** should be filled and signed by all group members and submitted to the tutor prior to each assessment stage (sprint). A copy of the Task Delivery Form can be found on BB and be downloaded. *Please ask for guidance if you need further clarifications regarding use of Task Delivery Form.*
3. ***Peer Assessment Form***, which can be found on BB, is required to be filled and submitted by everybody.
4. You may assume the system will accept cookies. No threading/concurrency considerations are required. All data must be stored and retrieved from the Java DB database, and the system should be accessible on localhost deployed on Glassfish server.
5. You will be supplied with one SQL script to build tables containing some samples of members, claims and payments. **(XYZ\_Assoc.sql**). This DB can be optimised or customised for better use, or to fit in the architecture, you setup. You must show that your database contains only the original data so that the data to be inserted as part of test cases can be distinguished.

Please note that each group member should be prepared to run any of the tasks and provide relevant explanations. You will be asked to show and explain your code - which must be readable and commented.