# WAGA Membership and Events System

APRIL 12, 2024

McWhirter Consultancy
Authored by: Jan Requiero, Denver Pilapil, Jordan Farrow and Juhn Ipapo

# Distribution list

| Who | What (hard/soft copy) | Version | When |
|---|---|---|---|
| Jordan Farrow | Soft Copy | 0.1 | 26/02/2024 |
| Denver Pilapil | Soft Copy | 0.1 | 26/02/2024 |
| Jan Requiero | Soft Copy | 0.1 | 26/02/2024 |
| Juhn Ipapo | Soft Copy | 0.1 | 26/02/2024 |

# Modification History

| Who | What | Version | When |
|---|---|---|---|
| Jordan | Drafted introduction | 0.1 | 2/03/2024 - 3/03/2024 |
| Jordan | Drafted scope | 0.2 | 4/03/2024 |
| Jordan | Added functional requirements | 0.3 | 16/03/2024 |
| Juhn, Denver, Jordan, Jan | Drafted Use Case diagram (external tool Lucidchart used) | 0.4 | 18/03/2024 |
| Juhn | Added to the distribution list | 0.5 | 19/03/2024 |
| Jordan | Modified Use Case diagram (created associations, use cases, modified design) | 0.6 | 22/03/2024 |
| Juhn, Jan | Expanded NFR and FR list | 0.7 | 24/03/2024 |
| Jordan | Expanded list of FR and assumptions, created glossary of terms | 0.8 | 25/03/2024 |
| Denver, Jordan, Juhn | Drafted Domain Model Class Diagram (external tool Lucidchart used) | 0.9 | 25/03/2024 |
| Juhn | Modified Use case diagram layout | 1.0 | 26/03/2024 |
| Jordan, Denver, Juhn, Jan | Revised NFR and FR list. | 1.1 | 26/03/2024 |
| Juhn | Created definition for the association within glossary of terms | 1.2 | 29/03/2024 |
| Jordan | Revised project scope | 1.3 | 7/04/2024 |
| Denver | Added Assumptions | 1.4 | 8/04/2024 |

| | | | |
|---|---|---|---|
| Jan | Revised introduction and scope | 1.5 | 8/04/2024 |
| Denver, Jordan | Expanded upon NFR list | 1.6 | 8/04/2024 |
| Jordan, Juhn, Jan | Edited introduction and glossary of terms | 1.7 | 8/04/2024 |
| Jan | Added on to work breakdown structure | 1.8 | 8/04/2024 |
| Juhn | Added subheadings to FR, NFR and assumptions | 1.9 | 8/04/2024 |
| Jordan | Drafted use case description | 2.0 | 22/04/2024 |

# Contents

# Introduction

WA Gymnast Association (WAGA) is an association with 250 active members, specializing in event hosting only accessible by its members. Currently, WAGA processes their core business functions manually through work performed by volunteers. The main functions include membership applications and event registration. Membership application forms are retrieved from the WAGA website for athletes to print and complete with a payment method attached (cheque, physical money, or credit card). The physical application form is posted to the secretary and individually input into the system on a stand-alone computer. To inform members of events, for instance, tests and competitions, the events manager would email each member a month prior to the event with the event details and a link to the external service, “Book Now” to apply for events and send payments for events. WAGA had noted that the external system is too cumbersome and wants to handle everything regarding events within the association. The new system will benefit the association by removing the manual processing of the business’ core functions, streamlining their processes through an online model. This online model will allow members to apply for memberships and manage their information at any time without the need for assistance from a volunteer. As a result, the manual processing required from volunteers will decrease. Furthermore, the association will become more independent as a custom solution replaces the reliance on third-party services, while improving the usability of event registration for users. Communication is more likely to improve between the users and the business as all active members will receive the same emails.

# Glossary of Terms

| Term | Definition |
|---|---|
| Visitor | Individuals which have not and currently do not have a membership with WAGA. |
| Non-member | Individuals which were once members however, have not renewed their membership subscription. |
| Member | Individuals which have an active membership subscription with the association. |
| WAGA | Acronym for 'WA Gymnast Association'. |
| Association | All volunteers within WAGA. |
| Athlete | The combination of visitor, non-member and member. |
| Event | An umbrella term consisting of both competitions and tests. |

# Project Scope

The system is expected to allow athletes to complete and submit a membership application form through the website with an online credit or debit card payment. The applications will be automatically processed, registering the user as a member and providing them with the ability to register for events within capacity. If necessary, payments for events will be handled through the website, similarly to membership applications. All members can submit an mp3 file with a maximum size of 10mb while applying for the competition. In addition, all email addresses of WAGA members will be automatically placed within a mailing list which allows the secretary and events manager to send out emails to all members. The system is to provide the association with the abilities to set the capacities of tests and export a list of members which have registered for an event, manage and update both the status of memberships and the data of its members and upload documents and articles to the website.

## Functional Requirements (FR)

3.1.1 The visitor must be able to apply for a membership subscription between January 1st and December 31st annually.

3.1.2 The member must be able to renew their membership subscription between January 1st and December 31st annually.

3.1.3 The non-member must be able to renew their membership subscription between January 1st and December 31st annually.

3.1.4 The visitor must be able to provide payment for a membership subscription.

3.1.5 The member must be able to provide payment for a membership subscription or event.

3.1.6 The non-member must be able to provide payment for a membership subscription.

3.1.7 The members must be able to manipulate their membership subscription details.

3.1.8 The non-member must be able to manipulate their membership subscription details.

3.1.9 The member must be able to login using unique credentials.

3.1.10 The non-member must be able to login using unique credentials.

3.1.11 The member must be able to register for events within capacity.

3.1.12 The member must be able to upload a music file for competitions.

3.1.13 The secretary must be able to send an email to the mailing list.

3.1.14 The secretary must be able to manipulate a membership subscription status between active and inactive.

3.1.15 The events manager must be able to send an email to the mailing list.

3.1.16 The events manager must be able to create events.

3.1.17 The events manager must be able to create a limit on the number of members which can register for a test.

3.1.18 The events manager must be able to export a list of members registered to an event.

3.1.19 The test convenor must be able to export a list of members registered to a test.

3.1.20 The secretary must be able to update data within a membership subscription.

3.1.21 The association must be able to upload documents and articles onto the website.

3.1.22 The system must be able to remove the non-members' email address from the mailing list.

3.1.23 The system must be able to add members to an external mailing list.

## Non-Functional Requirements (NFR)

3.2.1 Files uploaded by the member must be a maximum size of 10MB.

3.2.2 Music files uploaded by the member must be of mp3 format.

3.2.3 System analysis documentation must follow ISO 29184.

3.2.4 Athletes must consistently receive emails within 3 hours of its sent time.

## Out of Scope Items

3.3.1 The system will not allow any other payment methods excluding credit and debit online payment.

3.3.2 The system will not provide support services for the athletes for instance, customer support or financial support.

3.3.3 The system will not remind members of upcoming registered events.

3.3.4 The system will not allow a member to register for multiple events within one transaction.

3.3.5 The system will not refuse a membership application within end of the year time periods.

3.3.6 The system will not carry over a membership onto the following year, regardless of membership commencement date.

# Assumptions

4.1 It is assumed all users engaging with the system are English speaking, therefore other languages will not be supported.

4.2 It is assumed that the association has a bank account, therefore making online payment possible.

4.3 It is assumed that the only payment method by the association is online credit and debit card.

4.4 It is assumed that once a membership subscription expires, all members become non-members.

4.5 It is assumed that all members are removed from the mailing list automatically on the 31st December.

4.6 It is assumed that membership subscriptions do not automatically renew.

4.7 It is assumed that a visitor and non-member can apply or renew their membership subscription any time between January 1st and December 31st.

4.8 It is assumed all users are capable of basic computer skills for instance, typing and searching skills

4.9 It is assumed that once December 31st passes, all membership subscriptions expire, and members become non-members.

4.10 It is assumed that all users have access to the internet to access the online system.

4.11 It is assumed that the system will operate in WAGA time zone for all scheduling purposes including events and application deadlines.

4.12 It is assumed that the test convenor runs the tests therefore does not set up events and can only export the list of registered members

4.13 It is assumed that the system and its functions will be always active and accessible.

4.14 It is assumed that during the membership application process, the visitor and non-member choose their membership plan.

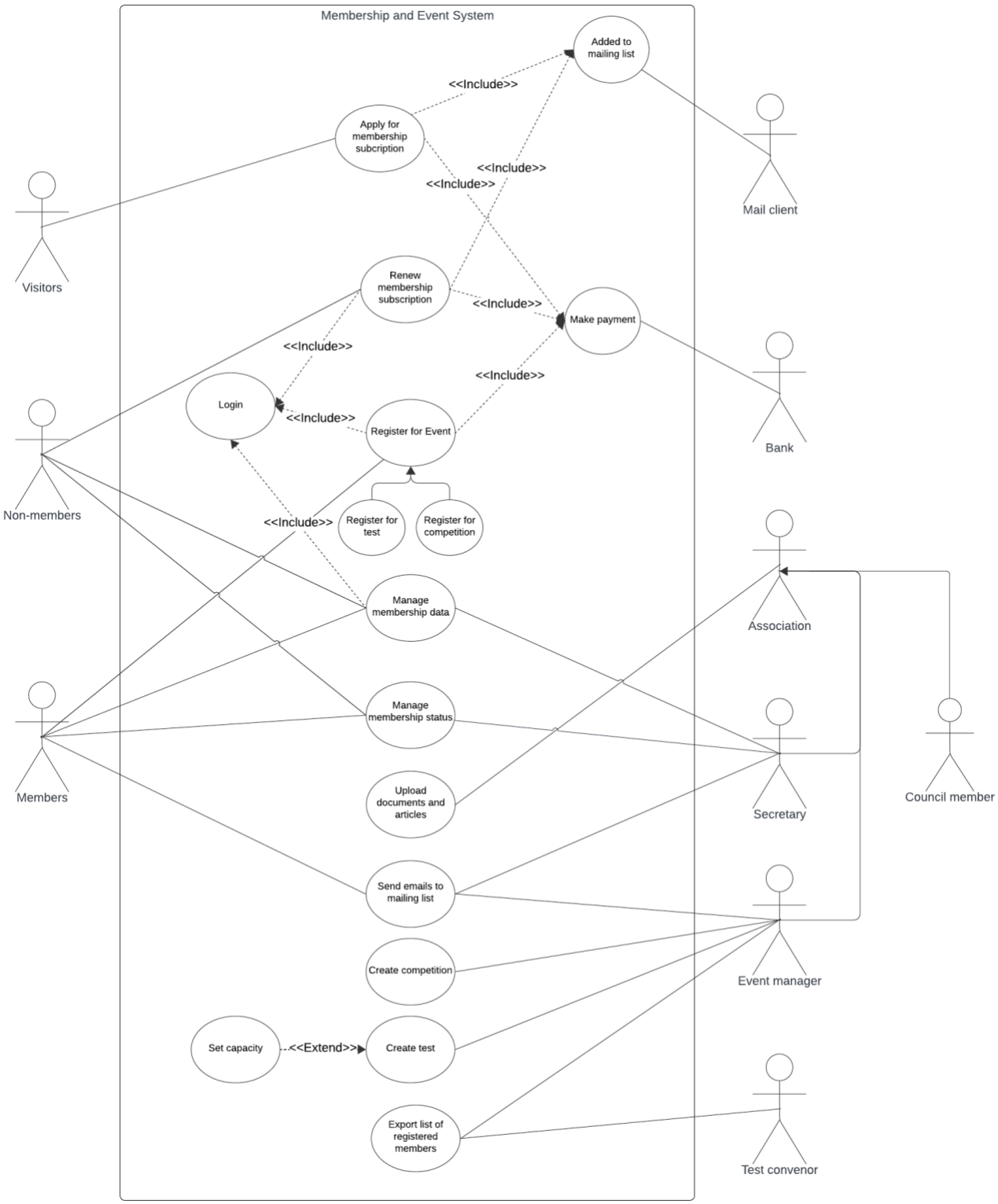
# Work Breakdown Structure (WBS)

| Task | Start Date | End Date | Responsible Team Member | Dependencies | Notes |
|---|---|---|---|---|---|
| *Project planning* | *March 2, 2024, 4pm* | *March 3, 2024, 7pm* | *Everyone* | *Case study* | *Kickoff meeting* |
| *Draft Scope* | *March 4, 2024, 2pm* | *March 5, 2024, 12pm* | *Jordan, Jan* | *Project Planning* | *Will need to be revised later.* |

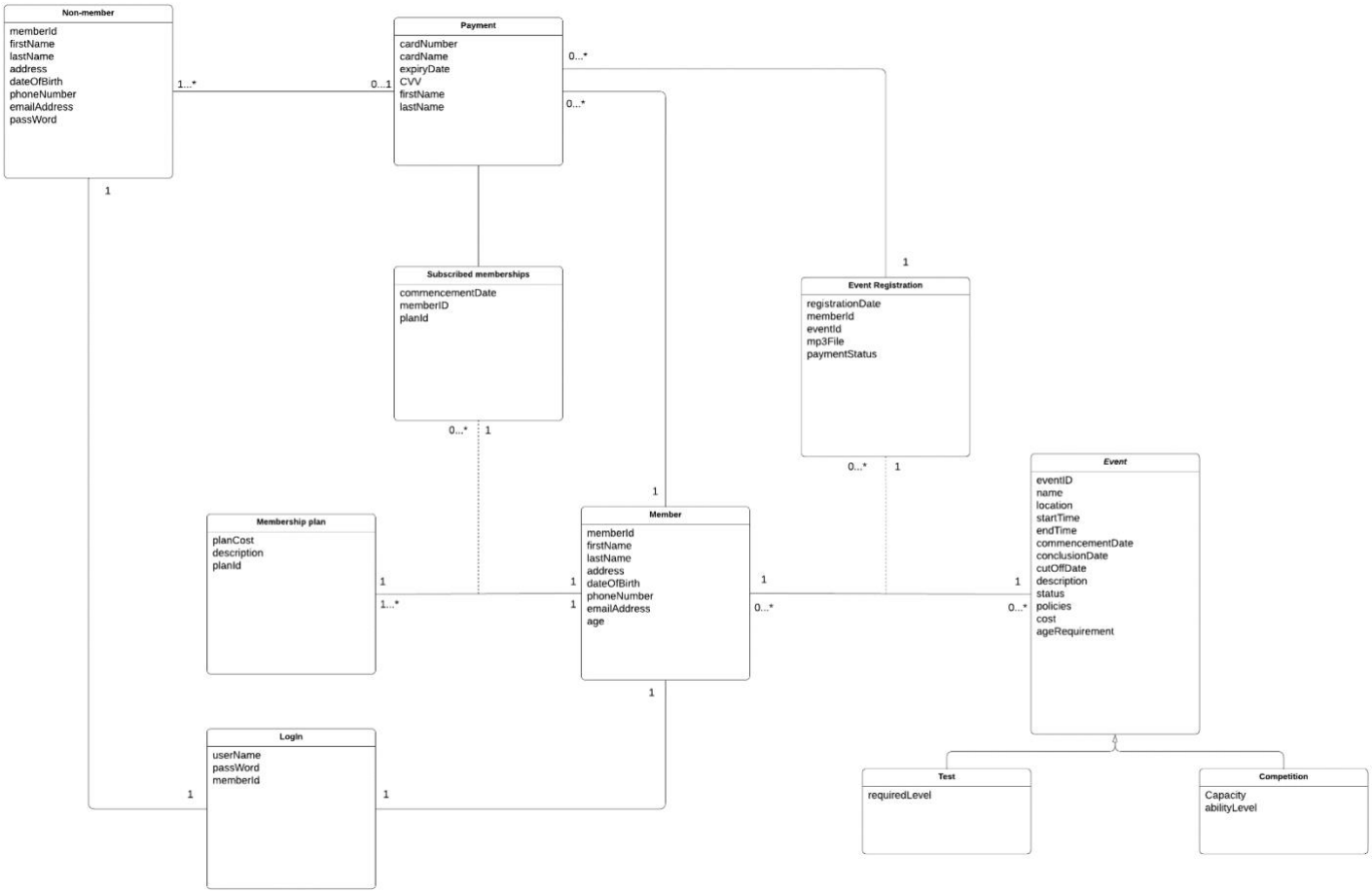
| | | | | | |
|---|---|---|---|---|---|
| *Meeting #1* | *March 5, 2025, 1:30pm* | *March 5, 2025, 2:35pm* | *All members* | *Case study* | *Two members missing.* |
| *Create Functional Requirements, Non-Functional Requirements List* | *March 16, 2024, 3pm* | *March 19, 2024, 11am* | *Jordan, Jan* | *Case study* | *FR & NFR list will change throughout time.* |
| *Meeting #2* | *March 18, 2024, 2pm* | *March 18, 2024, 2:30pm* | *All members* | *Report document, Functional and Non-Functional Requirements* | *One member missing.* |
| *Draft Use Case Diagram* | *March 18, 2024, 5pm* | *March 21, 2024, 12pm* | *Juhn, Denver, Jordan, Jan* | *Functional and Non-Functional Requirements* | *none* |
| *Distribution List* | *March 19, 2024, 10am* | *March 19, 2024, 11am* | *Juhn* | *Project planning* | *none* |
| *NFR and FR List review* | *March 24, 2024, 1pm* | *March 27, 2024, 4pm* | *Juhn, Jordan, Jan* | *Functional Requirements* | *none* |
| *Meeting #3* | *March 25, 2024, 1:30m* | *March 25, 2024, 2:30pm* | *All members* | *One member missing* | *One member missing* |
| *Glossary of Terms* | *March 25, 2024, 3pm* | *March 29, 5pm* | *Jordan, Jan* | *Project Planning* | none |
| *Draft Domain Model Class Diagram* | *March 25, 2024, 6pm* | *April 8, 2024, 1pm* | *Juhn, Jordan, Denver, Jan* | *Use Case Diagram* | none |

| | | | | | |
|---|---|---|---|---|---|
| *Meeting #4* | *April 8, 2024, 12:30pm* | *April 8, 2024, 2:30pm* | *All members* | *Project document* | *All members present.* |

# Use Case Diagram



# Domain Model Class Diagram



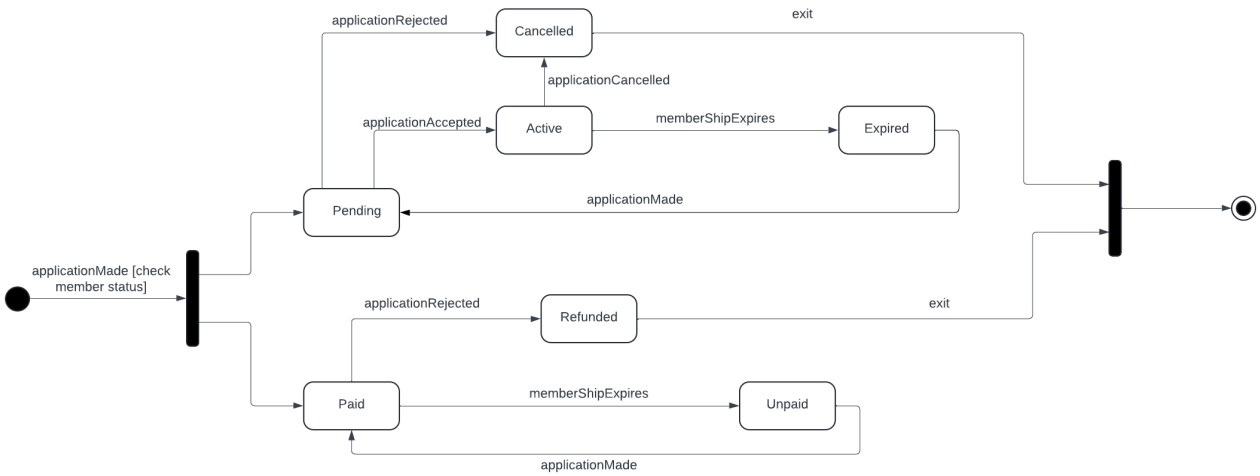

# End of Assignment 1

# State Machine Diagrams

Since assignment 1, I have expanded my class 'Subscribed Membership' to include objects paidStatus and membershipStatus which is incorporated within state machine diagram 1.

Since assignment 1, I have expanded my class 'Event Registration' to include object registrationStatus which is incorporated within state machine diagram 2.

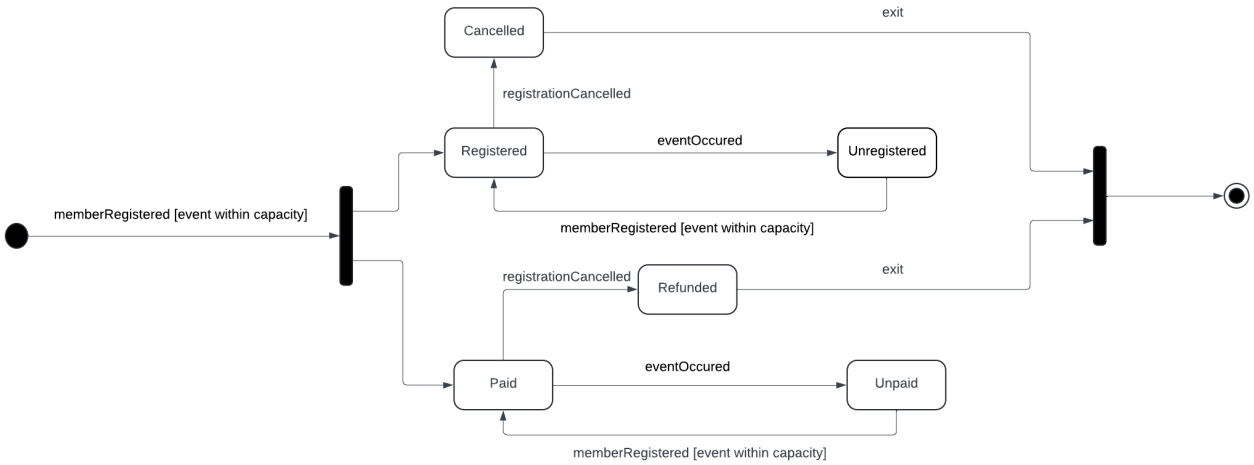
## State Machine Diagram 1 - Subscribed Memberships Class




The 'Subscribed Memberships' class begins within an open state until a visitor creates an application. The guard condition ensures that the visitor is not currently a member, otherwise both membershipStatus and paidStatus would already be within active and paid states. Once the application is made, the state of paidStatus becomes paid and membershipStatus becomes pending as the visitor awaits confirmation from the secretary. From pending, memberStatus can become one of two states, cancelled, which occurs when the secretary rejects the application or active, which occurs when the secretary accepts the application. When the state membershipStatus is active, the state of paidStatus is paid. When the membership expires, the state of membershipStatus changes to expired as they are a previous member and the state of paidStatus becomes unpaid as they now do not have a subscription which is paid for. The actor can remain within this state, therefore not having an active subscription or can renew their membership subscription. When a renewal application is made, the

state of membership status and paid status again become pending and paid respectively. In addition to membershipStatus being cancelled when the application is rejected, the state of paidStatus becomes refunded as the visitor receives the money spent on applying for a membership.

## State Machine Diagram 2 – Event Registration Class




The ‘Event Registration’ class begins within an open state until a member registers for an event. The guard condition ensures that there are available positions within the event. A competition will always satisfy this condition however, a test may be at capacity. Once the registration for an event is confirmed, two object’s states begin to change in parallel from the same transitions. The state of registrationStatus becomes registered, while the state of paymentStatus becomes paid. If a member chooses to cancel their registration, registrationStatus changes to cancelled and paymentStatus changes to refunded. As a result, the member has been refunded the costs of registering for the event and can no longer attend. Both objects then exit the sequence as there are no other possible states it can possess. If the event has passed, the member is no longer able to attend and therefore is deregistered. As a result, the states of registrationStatus becomes unregistered and paymentStatus becomes unpaid as they have no longer paid for an upcoming event. Both objects can remain within these states until the member chooses to register for another event. If a member chooses to register for another event, the guard condition must return true to ensure capacity has not been exceeded, otherwise the objects remain within its states of Unregistered and unpaid. If the member is able to register for an event within capacity, both registrationStatus returns to a registered state and paymentStatus returns to a paid state.

# Use Case Descriptions

## Elaboration of Apply for Membership Subscription Use Case

| | |
|---|---|
| **Use case name:** | Apply for membership subscription |
| **Triggering event:** | A visitor initiates the membership application process to become a member. |
| **Brief description:** | A visitor applies for a membership subscription, providing their personal and payment information, which are approved by the system, allowing the bank to process the payments and for the application to be forwarded to the secretary for approval. |
| **Actors:** | Visitor, Bank, Secretary |
| **Related use cases:** | Make payment |
| **Pre-conditions:** | Applications must be available.<br><br>Visitor applying for the membership subscription does not have an active account within the organization. |
| **Post-conditions:** | Visitor is an active member of the association, having access to event registration. |

| | | |
|---|---|---|
| | The now member's information is saved with an assigned membership identification string. | |
| **Flow of activities:** | **Actor** | **System** |
| | 1. The visitor requests application form | |
| | | 1.1. System presents application form |
| | | 2. System prompts for personal details and plan type |
| | 2.1. Visitor inputs personal details and plan type | |
| | | 2.2 System validates personal information and plan type |
| | | 3. System prompts for payment information |
| | 3.1. Visitor enters payment information | |
| | | 3.2. System validates payment information |
| | | 4. System forwards payment information to bank |
| | 4.1. Bank approves payment | |
| | 4.2. Bank forwards confirmation to system | |
| | | 5. System forwards application form to secretary |

| | | |
|---|---|---|
| | | 6. Membership identification is created. |
| | | 7. Personal information, membership identification and plan type are saved. |
| | IF visitor chose to save payment information with system, go to step 7.1.<br>Else, go to step 8. | |
| | | 7.1 Payment information is saved. |
| | | 8. Outputs confirmation message |
| **Exception Conditions** | 2.2 Personal details are invalid, incorrect, or not fulfilled, so visitor must be re-input personal details at step 2.1<br><br>2.2 Plan chosen is unavailable or not selected, so visitor must re-input a plan type at step 2.1.<br><br>3.2 Payment information fields are incorrect or invalid, therefore visitor must re-input payment information at step 3.1.<br><br>4.1 Insufficient funds are present within account or account is invalid, therefore payment is not deducted, so visitor must re-input payment information at step 3.1. | |

## Elaboration of Register for Competition Use Case

| | |
|---|---|
| **Use case name:** | Register for a competition |
| **Triggering event:** | A member initiates the competition registration process. |
| **Brief description:** | A member registers for an event using their membership information, providing payment if required. The payment is processed by the bank and the member's registration and required information is saved. |
| **Actors:** | Member, Bank |
| **Related use cases:** | Make payment |
| **Pre-conditions:** | Individual must be an active member with a subscription.<br><br>Member must be authenticated through login.<br><br>Registration for the competition must be available.<br><br>Member must be within the requirements of the competition for instance, between minimum and maximum ranking. |
| **Post-conditions:** | A member is registered for a competition. |

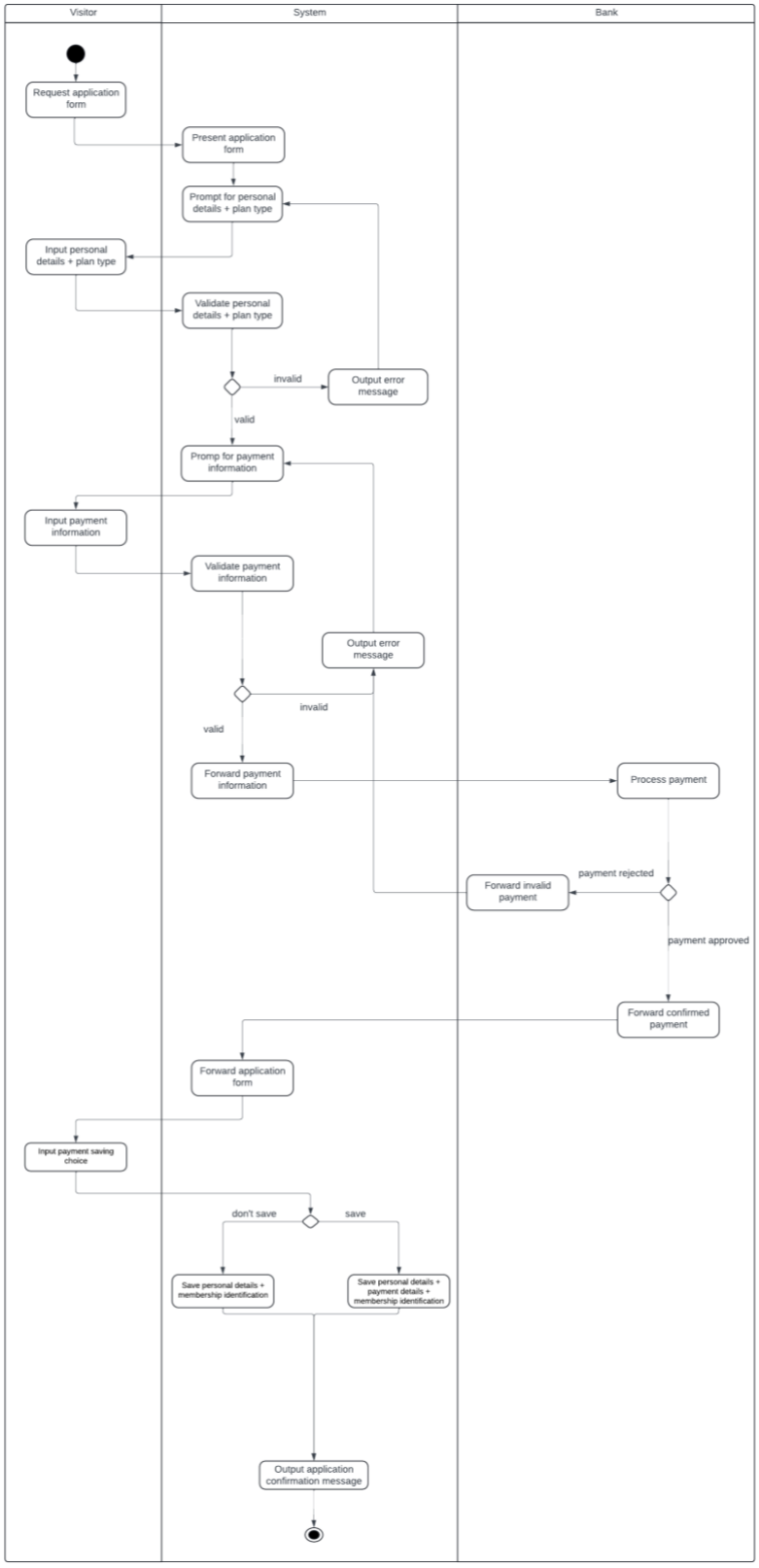
| | | |
|---|---|---|
| | A music file is validated and saved.<br><br>Payment if required, has been processed. | |
| **Flow of activities:** | **Actor** | **System** |
| | 1. Member requests competition registration form | |
| | | 1.1 System presents competition registration form |
| | | 2. System prompts for membership information |
| | 2.1. Member inputs membership information | |
| | | 2.2. System validates membership information |
| | | 3. System prompts for music file |
| | 3.1. Member inputs music file | |
| | | 3.2. System validates music file |
| | IF payment not required for competition registration, go to step 5.<br>Else, continue. | |
| | IF payment information was not saved within system, continue.<br>ELSE, go to step 4.2. | |
| | | 4. System prompts for payment information |

| | | |
|---|---|---|
| | 4.1 Member enters payment information | |
| | | 4.2 System validates payment information |
| | | 5 System forwards payment information to bank |
| | 5.1 Bank approves payment | |
| | 5.2 Bank forwards confirmation to system | |
| | | 6. System saves membership identification for the competition |
| | | 7. System outputs confirmation message |
| **Exception Conditions** | 1. Registration for the competition is not yet available or had passed, therefore access to the competition registration process is denied.<br><br>2.2 Membership information was incorrect or invalid, therefore the member must be redirected to step 2.1.<br><br>3.2 The music file is not of mp3 format and/or is above or below 10mb, therefore is invalid, so the member must be redirected to step 3.1.<br><br>4.2 Payment information fields are incorrect or invalid, therefore the member must be redirected to step 4.1. | |

| | |
|---|---|
| | 5.1 Insufficient funds are present within account or account is invalid, therefore payment is not deducted, so visitor must re-input payment information at step 4.1. |

# Activity Diagrams

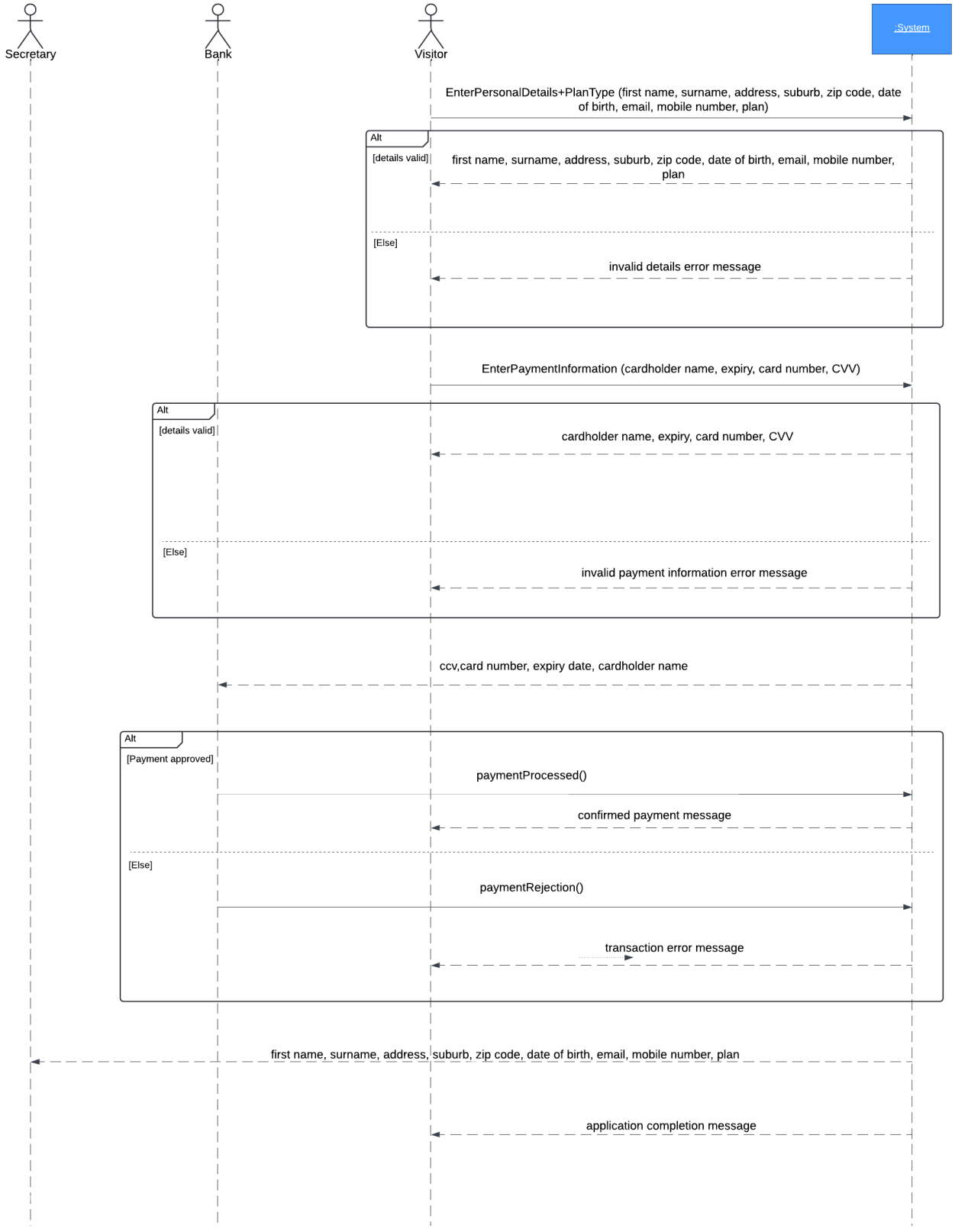
## Activity Diagram 1 Apply for Membership Subscription



The *Apply for Membership* use case begins in an open state until the visitor requests the membership application form from the system. The system responds by presenting the form and prompting the visitor for their personal details and choice of membership plan. The visitor enters their personal details and plan type which are validated by the system. If the system deems either the fields for the personal details and/or plan type are empty or invalid, the system outputs an error message to the visitor and re-prompts the for an input of personal details and plan type. If it was valid, the system then prompts the visitor for their payment information. The visitor inputs the required information which is again validated by the system. If there are fields which are empty or invalid, the system will output an error message and re-prompt the user to input their payment information. If valid, the system forwards the payment information to the bank which will process the payment. If the bank cannot process the payment due to insufficient funds or invalid account details, the bank will forward an invalid payment message, which prompts the system to output an error message and re-prompt the visitor to input their payment information. If there are sufficient funds, the payment is processed. The bank then forwards the confirmation of payment, causing the system to forward the application form to the secretary. The system then prompts the visitor with the option to save their payment information. If the visitor chooses to not save their payment information, the system will only save the visitor's personal details. If the visitor chooses to save their payment information, the system will save the visitor's personal and payment information. The process ends with the system outputting a application confirmation message.

# Systems Sequence Diagrams

## Sequence Diagram 1 Apply for Membership Subscription



The visitor inputs their personal details and their chosen plan type. If the system deems the input valid, it echoes the information back to the visitor, else, it outputs an error message regarding the invalid input. The user then inputs their payment information. If the system deems the input valid, the input is echoed back to the visitor, else, an error message regarding the invalid payment information will be output. The valid payment information is then output to the bank, which will process the payment. If the payment is approved, the bank inputs a processed payment to the system, allowing the system to output an approved payment message to the visitor. If it cannot be processed, a rejected payment is input to the system, allowing the system to output a rejected payment message to the visitor. Then, the valid personal and payment information are output to the secretary, which will then decide to input an approval or rejection of the application. Finally, the system outputs a message notifying the visitor that the application form has been completed and submitted.

# User Interface (GUI)

Since assignment 1, I have expanded on my assumptions list to include, ‘It is assumed the organization will have an image to use for the final product”. As a result, IMAGE will be used as a placeholder to signify the location of the image.

## User Interface 1 – Apply for Membership Subscription

# Data Dictionary

## Elementary Dictionary (Apply for Membership Subscription)

| Legend |
|---|
| B: button |
| CB: check box |
| D: date |
| L: links to another section of website |
| N: any positive single digit |
| P: pull down menu |
| R: radio button |
| X: any keyboard character |

| Name | Type | Size | Format | Constraint |
|---|---|---|---|---|
| address | X | 20 | | Address must begins with its corresponding digit |
| about-us-link | L | 8 | | Name="About Us" |
| back-button | B | 6 | | Name="> Back" |
| cancel-button | B | 6 | | Name="Cancel" |
| cardholder-name | X | 35 | | |
| card-number | X | 16 | NNNNNNNNNNNNNNN | |
| confirm-button | B | 7 | | Name="Confirm" |
| contact-us-link | L | 10 | | Name="Contact Us" |
| cvv | X | 3 | NNN | |
| date-of-birth | D | 10 | dd/mm/yyyy | |
| email | X | 25 | | |
| events-link | L | 6 | | Name="Events" |
| expiry | D | 5 | mm/yy | |
| first-name | X | 20 | | |
| gender-choice | X | 1 | R | One of male, female, or other |
| image-link | L | 1 | | |
| members-link | L | 7 | | Name="Members" |
| mobile-number | X | 10 | NNNNNNNNNN | Must begin with '0' |
| plan-choice | X | 1 | R | One of plan |
| save-payment-choice | X | 1 | CB | One of save payment information |
| suburb | X | 15 | | |
| surname | X | 15 | | |
| zip-code | X | 4 | NNNN | |

## Composite Data Dictionary

web-page = image-link + about-us-link + events-link + contact-us-link + members-link + back-button + first-name + surname + date-of-birth + gender-choice + address + suburb + zip-code + email + mobile-number + plan-choice + cardholder-name + expiry + card-number + save-payment-choice + cvv + confirm-button + cancel-button

# References

(if any)