

Università della Calabria
Corso di Laurea Magistrale in
Statistica ed Informatica per le Decisioni
e le Analisi di Mercato
Anno Accademico 2019-2020

Progetto del corso di

Sistemi Informativi, Modulo B:
Progettazione di Sistemi Informativi

Information System Development for a Music Streaming Platform

Docente
Francesco Parisi

Studenti

Pietro Ruffo matr 214241
Othman Mohammed Al-Akhali matr 214795
Marco Tucci matr 214216

Summary

- Introduction..... 3
- 1 Requirements Analysis 4
 - 1.1 Scenario Analysis 4
 - 1.2 Specification of Requirements..... 8
- 2 Design..... 29
 - 2.1 Database Design 29
 - 2.1.1 Conceptual Model 30
 - 2.1.2 Relational Model..... 32
- 3 Implementation 36
- Appendix 37
 - Glossary of Terms 38
- Attached files..... 40

Introduction

Omegastream is an Italian media provider and music streaming service.

The company's business consists of providing an audio-streaming platform. Since it is a web-based firm, the target audience is millennials and in general people between 15 and 25 years old.

The company's main source of income comes from Advertisement and User Subscription. However, it is trying to raise funds from external investors attracted by the performing results of such business.

The company consists of the following departments: Software Development, Data Analysis, Support, and Marketing.

The internal organization has a flat structure with all the departments working on the same level, however, strategic decisions are made within the Marketing Department which plays an important role as coordinator.

The Marketing Department works on promoting the app to users in order to download it and subscribe. Both normal and premium users enjoy the same functionalities of support, feedback and listening to songs, although the premium users have full access to all the songs without ads. The organization has a Data Analysis team that collects data from both the internal and external environment to support Marketing in taking data-driven decisions.

The Information System needed for the above-mentioned business has to be able to:

1. Make possible the platform usability (Search for a Song and Listen to It). This functionality gives the user the ability to find any song easily, this function could enhance the quality of their experience and brand loyalty.
2. Support the Data Analysis Department in completing a Churn Analysis Report (Churn Analysis Report Completing). This functionality helps the Marketing Department to predict and reduce churn by making data-driven decisions.
3. Improve the customer experience (Bug Management and Bug Fixing). These functionalities allow collecting all the feedback from clients helping the development team to fix the bugs and upgrade the software to cope up with the user's needs.

1 Requirements Analysis

1.1 Scenario Analysis

The business process is carried out through the following steps: the Marketing Department promotes the platform on its social media channels; through the advertisement the visitor is brought to the company's webpage where they are invited to subscribe to the newsletter through which they will receive news about their favorite artists, media contents and discounts for the premium version subscription. The main banner will let them download the software for free. The process is basically the same on the smartphone, however, from the website instead of a banner, there will be a link to *Google Play* or *Apple Store* where it will be possible to download the free app.

Advertisement represents the first source of income for the business.

Once signed up the new free user will enjoy the premium functionalities for 30 days before deciding either to buy the premium version or to keep the free one.

The free version of the software can be used as a promotion channel by external companies, meaning that other firms can pay to promote their products or services on the streaming platform.

The free version allows the user to access most of the catalogue of songs and they will have to listen to advertisement campaigns, plus banners will appear on the free user interface. During the free experience, the advertisement for the premium version is made through the newsletter and audio campaigns on the platform.

The goal is to convert and then maintain as many free users as possible. After a period of usage, the free user subscribes to the premium version which offers a more complete experience, with the full catalogue and without advertisement.

Both free and premium customers can use a specific support section on the platform in order to report bugs and complaints, or give suggestions to improve the product. This is also aimed to build a community where it is possible for all the users to interact as well as share feedback to improve the software.

All this data coming from all the departments as well as user interactions with the platform are collected and processed by the Data Analysis Department that will carry a churn analysis and complete a report that will be sent to the Marketing Department which will use it in order to predict and prevent churning by making the strategic decisions needed to keep premium users as long as possible within the customer base. Once the report has been examined from the Marketing Department, they will either design a new promotion campaign or request new functionalities to the Software Development Department in order to cope up with future trends.

The functionalities required by the Information System are:

1. Make possible the platform usability (Search for a Song and Listen to It);
2. Support the Data Analysis Department in completing a Churn Analysis Report (Churn Analysis Report Completing);
3. Improve the customer experience (Bug Reporting and Bug Fixing).

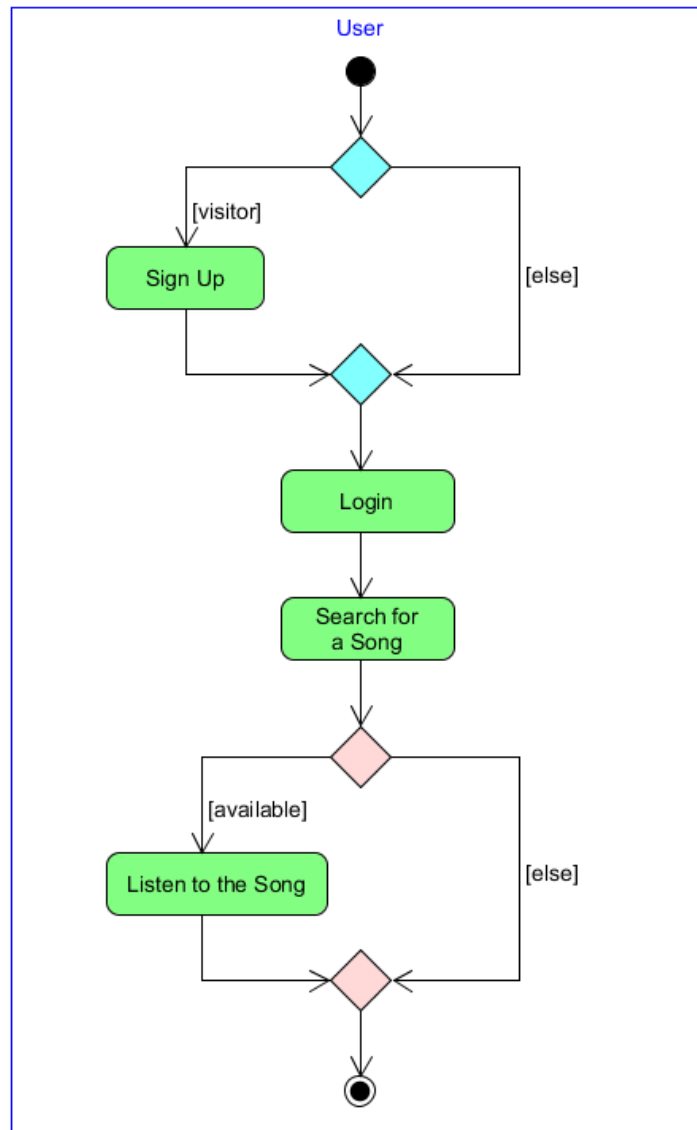
1.1.1 Make possible the platform usability

This functionality gives the user the possibility to find any song easily, this is why this process has to be as smooth as possible in order to increase the quality of the experience and thus, their loyalty to the platform. This procedure is made up of four macro-activities carried out by the User: Sign up, Login, Search for a Song and Listen to the Song.

First the user has to connect their device to the Internet, then open the application and access the platform. Having done that, the visitor will sign up if not registered yet, or directly login if an account was already created. Once completed this procedure, the interface will offer the user the possibility to search for a song among those available. The user will search for a song; thus, the platform will have to check the availability with regards to its actual presence in the catalogue as well as the possibility of listening to it which is subordinate to have either a premium or a free account.

Next, if the song is available, the user will play and listen to it, otherwise, the user will receive a notification of unavailability where they will be invited to search again for a different title.

The UML activity diagram below shows how the process is executed.

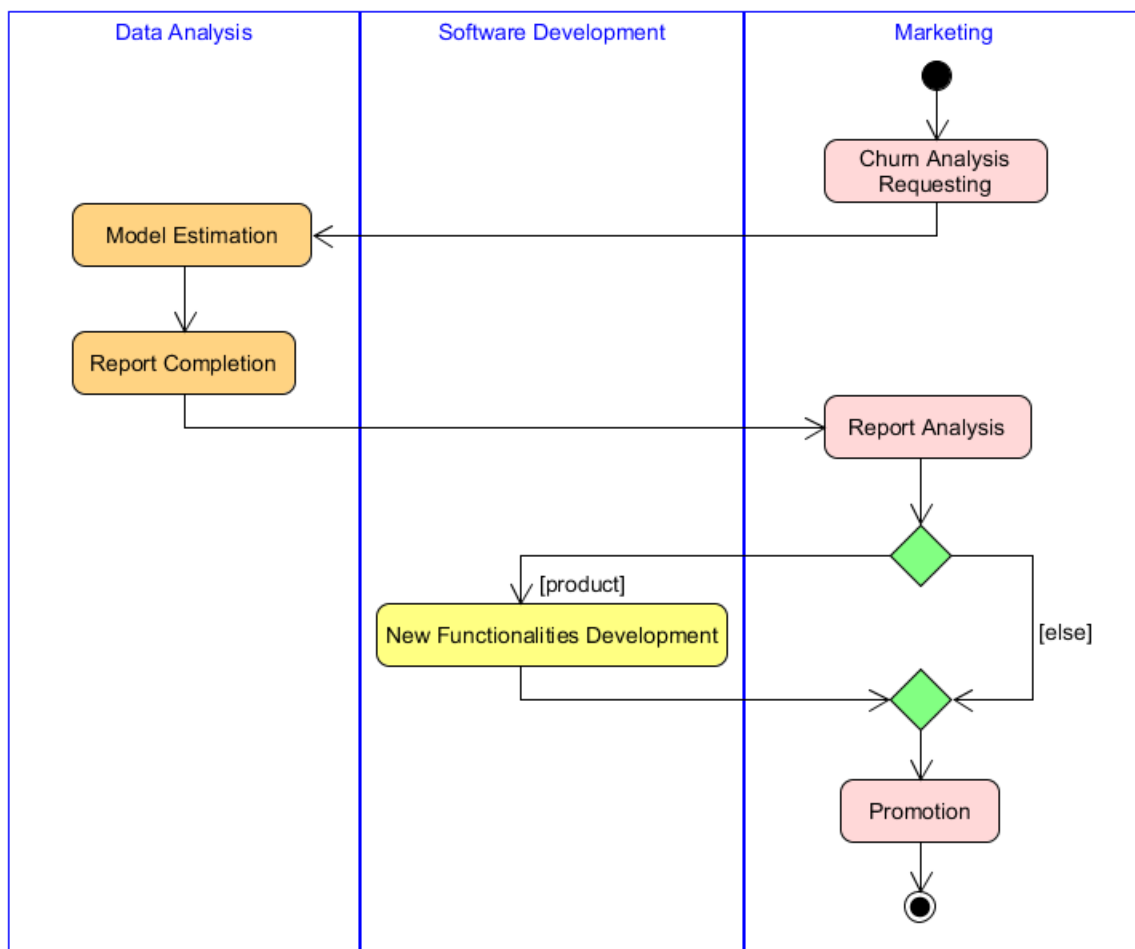


1.1.2 Churn Analysis Report Completing

This functionality is executed within the internal organization and allows Omegastream to predict and reduce churning by taking the right decisions in order to increase users' loyalty to the platform. This process is carried out through six macro-activities: Churn Analysis Requesting, Model Estimation, Report Completion, Report Analysis, New Functionalities Development and Promotion. Those activities are performed by the following departments: Data Analysis, Software Development and Marketing.

This process starts with the Marketing Department formally requesting the Data Analysis team a Churn Analysis. When the request is received, the Data Analysis Department starts working on it. Once the best model has been estimated, a report will be carried out and then sent to the Marketing Department.

At this point of the process, the Marketing analyzes the churn analysis report and decides whether actions will be taken on either the product side by implementing new functionalities or by just promoting the platform differently thus applying a different pricing or branding policy.



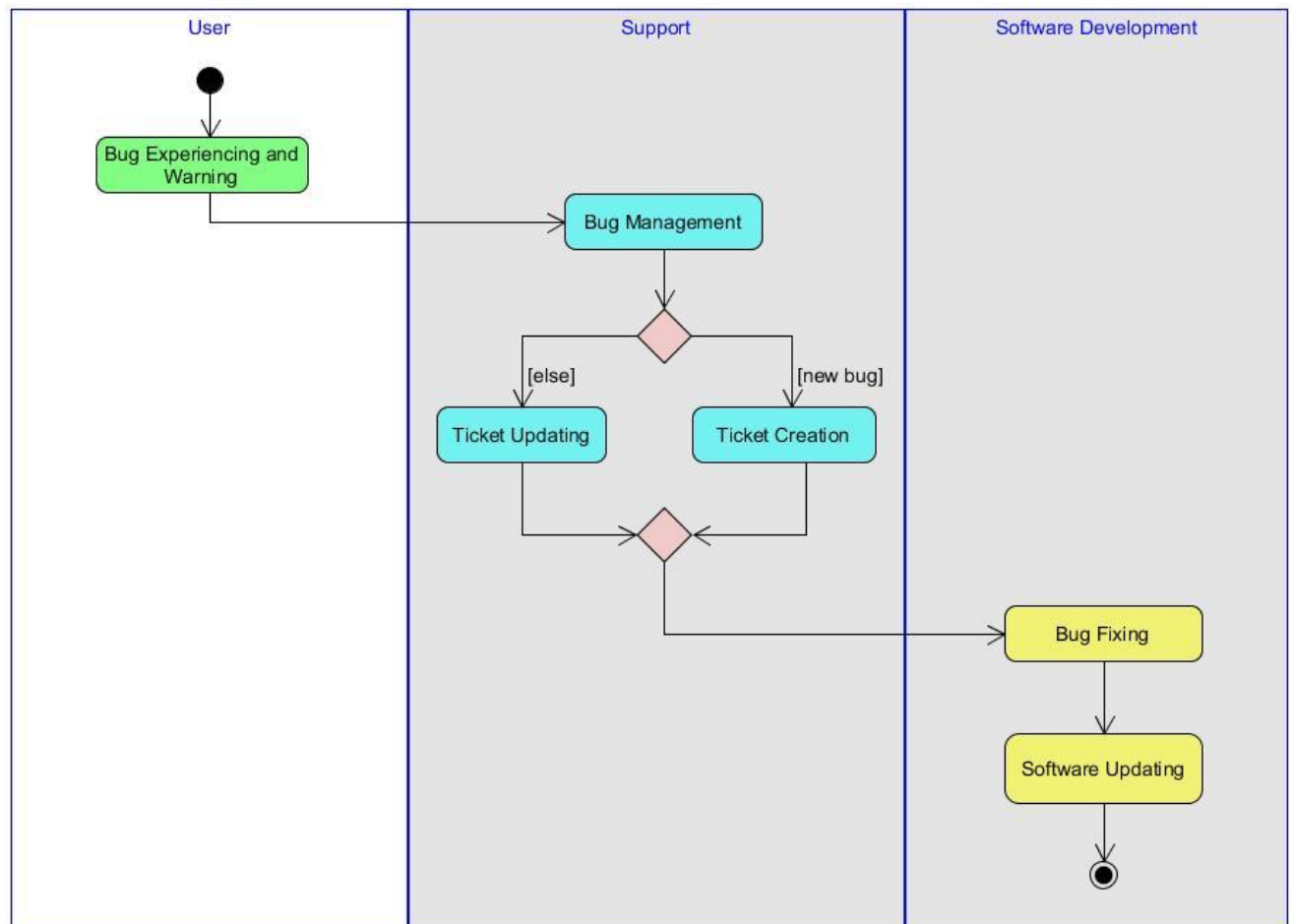
1.1.3 Bug Reporting and Bug Fixing

Through this application all the feedback from clients are collected in order to help the Development team to fix the bugs and upgrade the software.

The process is made up of six macro-activities: Bug Experiencing and Warning, Bug Management, Ticket Updating, Ticket Creation, Bug Fixing and Software Updating.

The above-mentioned activities are executed by the User, the Support Department, and the Software Development Department.

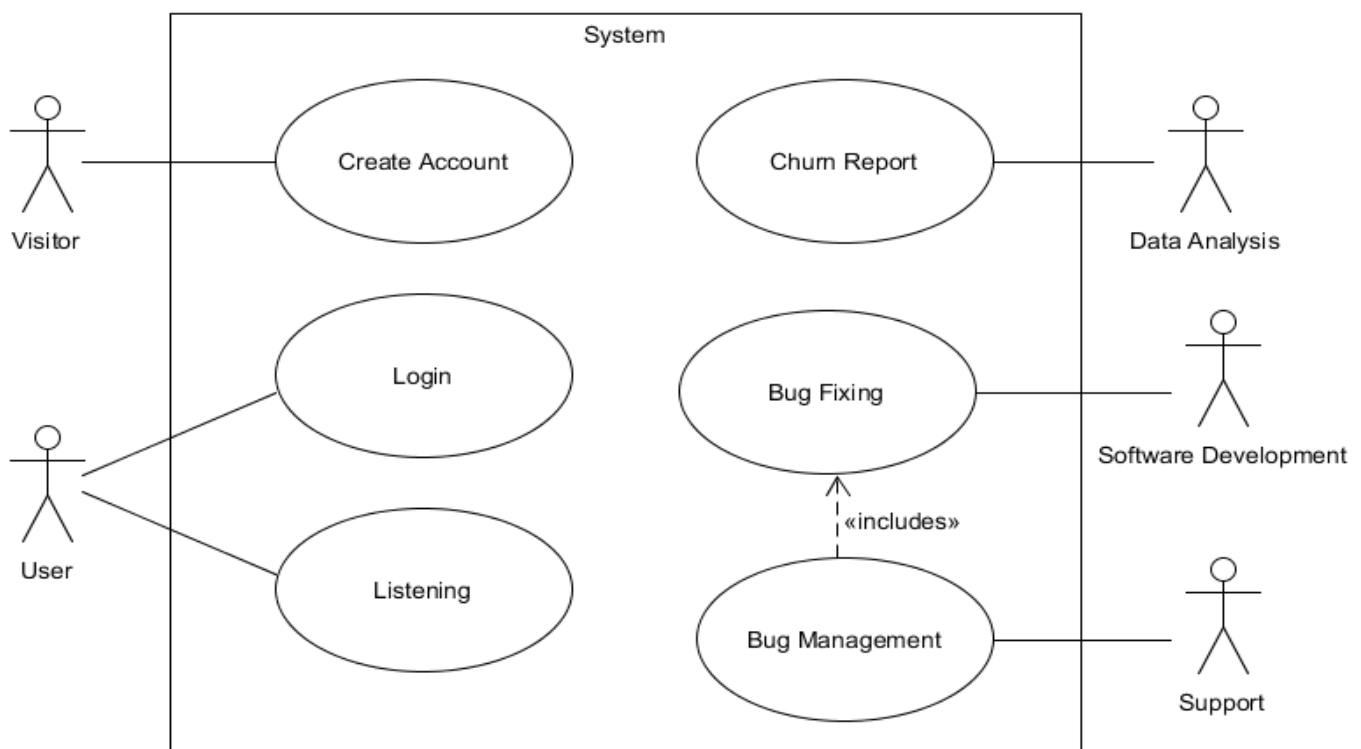
The user experiences the bug and warns the Support which will immediately take charge of it. Having done that, the Support will check whether it is a bug already experienced from other users or if it is a completely new one. In the last case, the Support will create a ticket, meaning a service request from an end-user. In case a ticket was already created for that specific bug, the Support will just update it. To conclude, the developers will take charge of the issue by fixing the bug and eventually updating the system. The UML activity diagram below shows how the process is structured with the grey swim lanes representing the internal environment of the company and the white one representing the exterior.



1.2 Specification of Requirements

The use case diagram below shows that there are 5 main actors interacting with the system through 6 use cases:

- The Visitor is involved in the “Create Account” use case;
- The User takes part in the following use cases: “Login” and “Listening”;
- The Data Analysis Department carries out the “Churn Report” use case;
- The Software Development Department is involved in the “Bug Fixing” use case;
- The Support Department takes part in the “Bug Management” use case which includes the “Bug Fixing” use case.



Aspects concerning Promotion could have been developed as well but for the moment they will be executed “manually” meaning outside the Information System.

Future updates and changes in the Information System will probably take those aspects into account depending on the needs of Omegastream.

“Create Account” Use Case

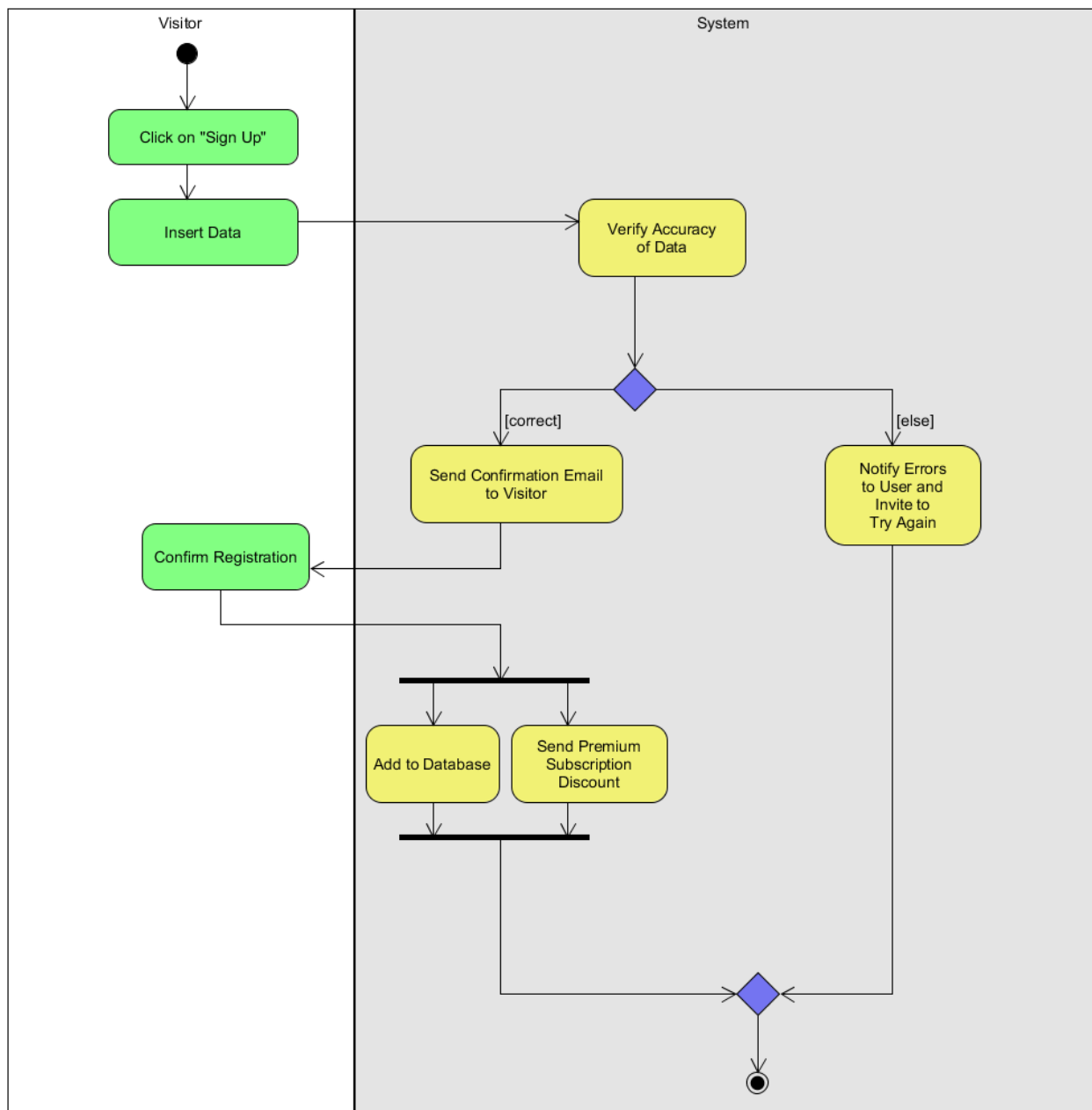
This use case is mainly focused on the visitors who open the platform and get to the registration page. This step requires the download of the platform on a device (computer, smartphone or tablet) and a valid email address that can be used to sign in. The actor of this use case is a visitor who is anyone opening the subscription link of the platform.

The visitor enters the details such as, email, password and personal data, the system verifies the data and confirms the registration, so the new user is added to the database and offered discounts.

“Create Account” Use Case – Table description

Name	Create Account
Description	It allows the visitor to sign up on the platform and create a user account so that they can log in and enjoy all the contents offered.
Basic flow	<ol style="list-style-type: none">1. The visitor clicks on “Sign Up”.2. The visitor inserts all the data (e.g. personal data, email, password, and username) requested by the system and then confirms.3. The system verifies the accuracy of the data and the presence of missing data.4. The system sends a confirmation email to the visitor.5. The visitor opens the link and confirms their registration.6. The system creates a new account by adding the new user’s data to the database.7. The system sends an email to the new user containing a discount for the premium subscription.
Alternative flows	<ul style="list-style-type: none">• At step 3 of the basic flow: Inaccurate and/or missing data<ol style="list-style-type: none">1. The system notifies the visitor which data are missing or inaccurate and invites them to try again from the beginning.
Pre-conditions	App downloaded on a smartphone, tablet, or computer.
Post-conditions	Success: the visitor creates a new account. Failure: the visitor does not create a new account.
Special requirements	Internet connection.

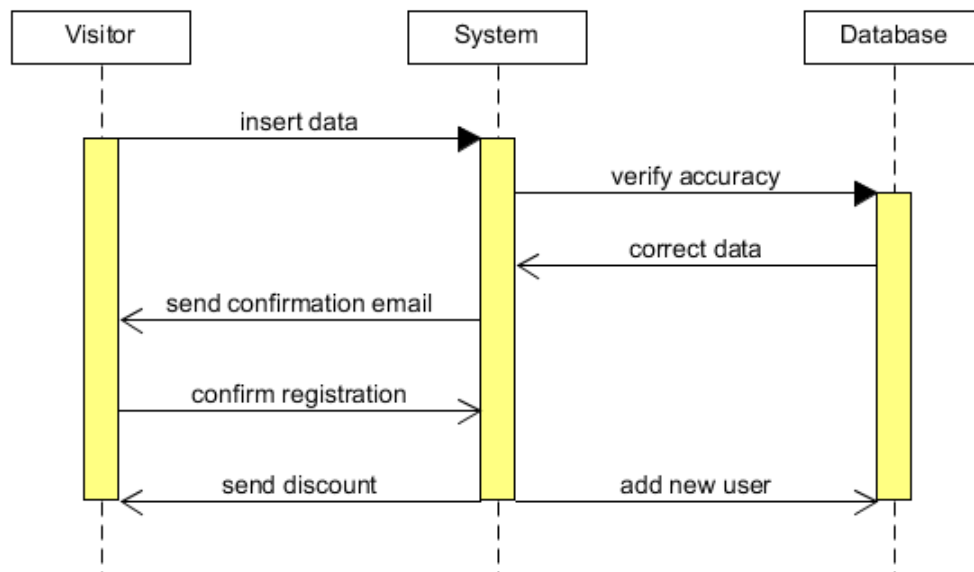
“Create Account” Use Case – Activity Diagram



Note: “Invite to Try Again” refers to a notification the visitor receives inviting them to start over the process from the beginning, meaning going back to “Click on Sign-up”.

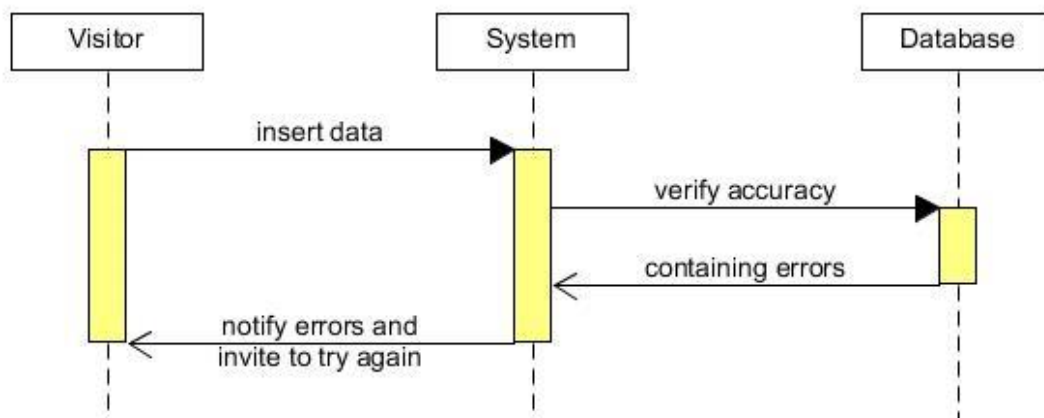
“Create Account” Use Case – Sequence Diagram (Basic Flow).

The account is successfully created.



“Create Account” Use Case – Sequence Diagram (Alternative Flow 1).

Inaccurate or missing data: the system notifies the visitor which data are missing or inaccurate (e.g. already existing email) and invites them to try again from the beginning.



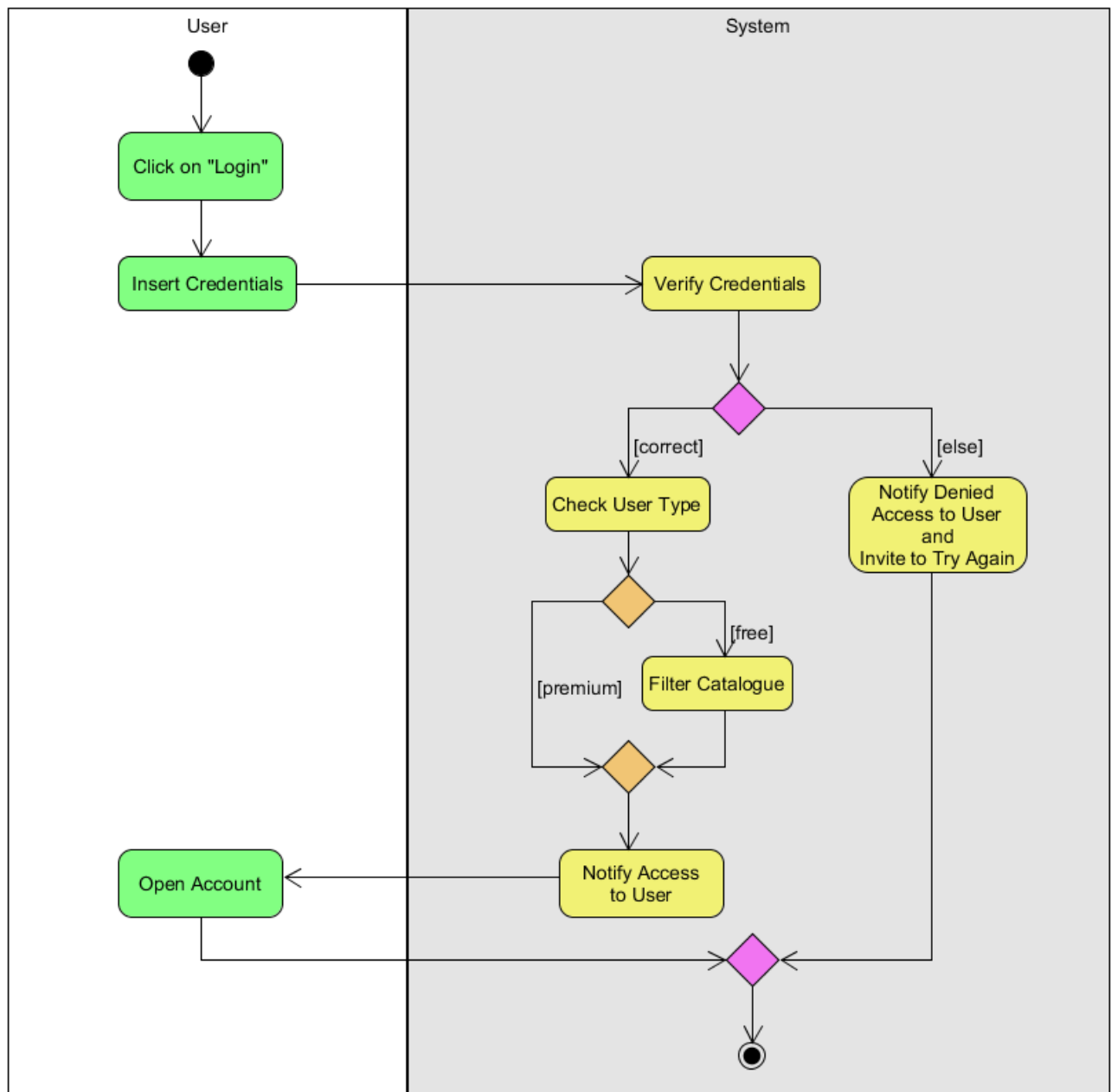
“Login” use case

This use case is performed by the user who login into an existing account in the platform. The user enters the credentials to be verified by the system that defines whether the user is of a premium account or a free one. The actor of this use case is the user who is a pre-registered customer that has a username and a password.

“Login” Use Case – Table description

Name	Login
Description	It enables the user to log in the platform so that they can search for a song and listen to it.
Basic flow	<ol style="list-style-type: none">1. The user clicks on “Login”.2. The user inserts their credentials (email, or username, and password) and confirms.3. The system verifies the accuracy of the credentials through the database.4. The system checks if the user is a premium account.5. The system notifies to the user they have successfully logged in.6. The user opens their account.
Alternative flows	<ul style="list-style-type: none">• At step 4 of the basic flow: The user has a free account<ol style="list-style-type: none">1. The system filters the catalogue of the songs through the database.2. Go to step 5 of the basic flow.• At step 3 of the basic flow: Wrong credentials<ol style="list-style-type: none">1. The system notifies the user that their access has been denied and invites them to try to login again, restarting the procedure from the beginning.
Pre-conditions	App downloaded on a smartphone, tablet, or computer.
Post-conditions	Success: the user logs in successfully. Failure: access denied.
Special requirements	Internet connection.

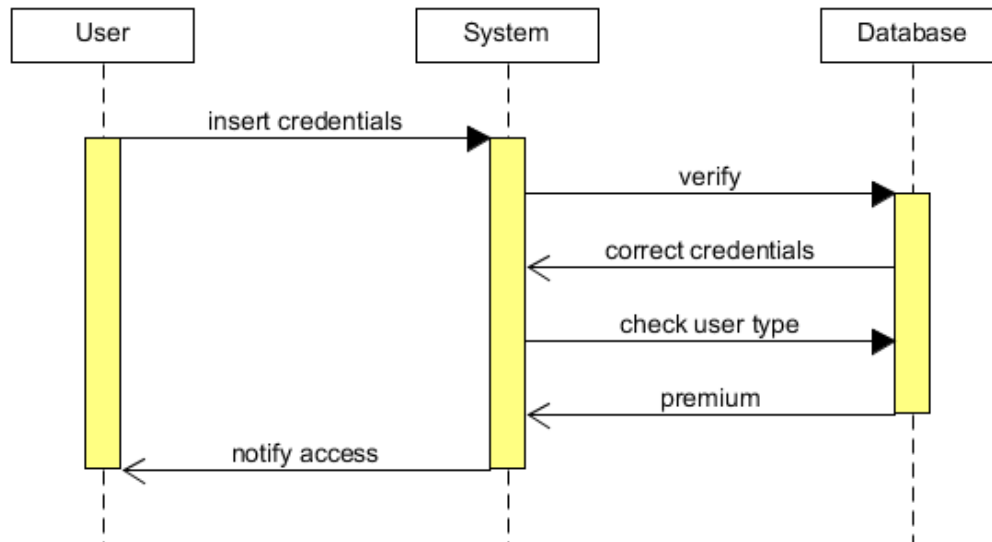
“Login” Use Case – Activity Diagram



Note: “Invite to Try Again” means that the user receives a notification inviting them to start over the process from the very beginning, that is clicking again on “Login”.

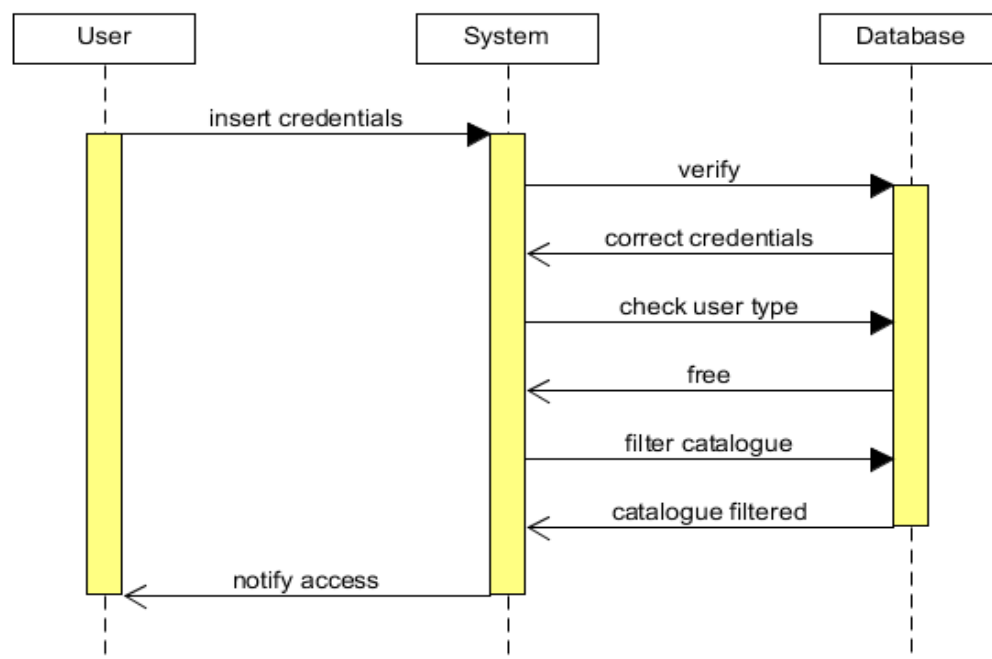
“Login” Use Case – Sequence Diagram (Basic Flow)

The premium user successfully logs in.



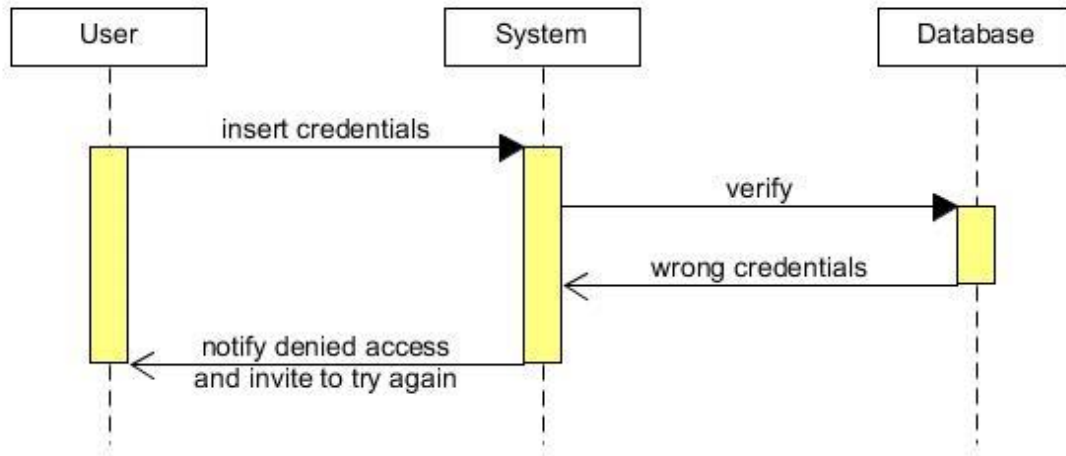
“Login” Use Case – Sequence Diagram (Alternative Flow 1)

The free user successfully logs in.



“Login” Use Case – Sequence Diagram (Alternative Flow 2)

Wrong credentials: the system notifies the user that their access has been denied and invites them to try to login again, restarting the procedure from the beginning.



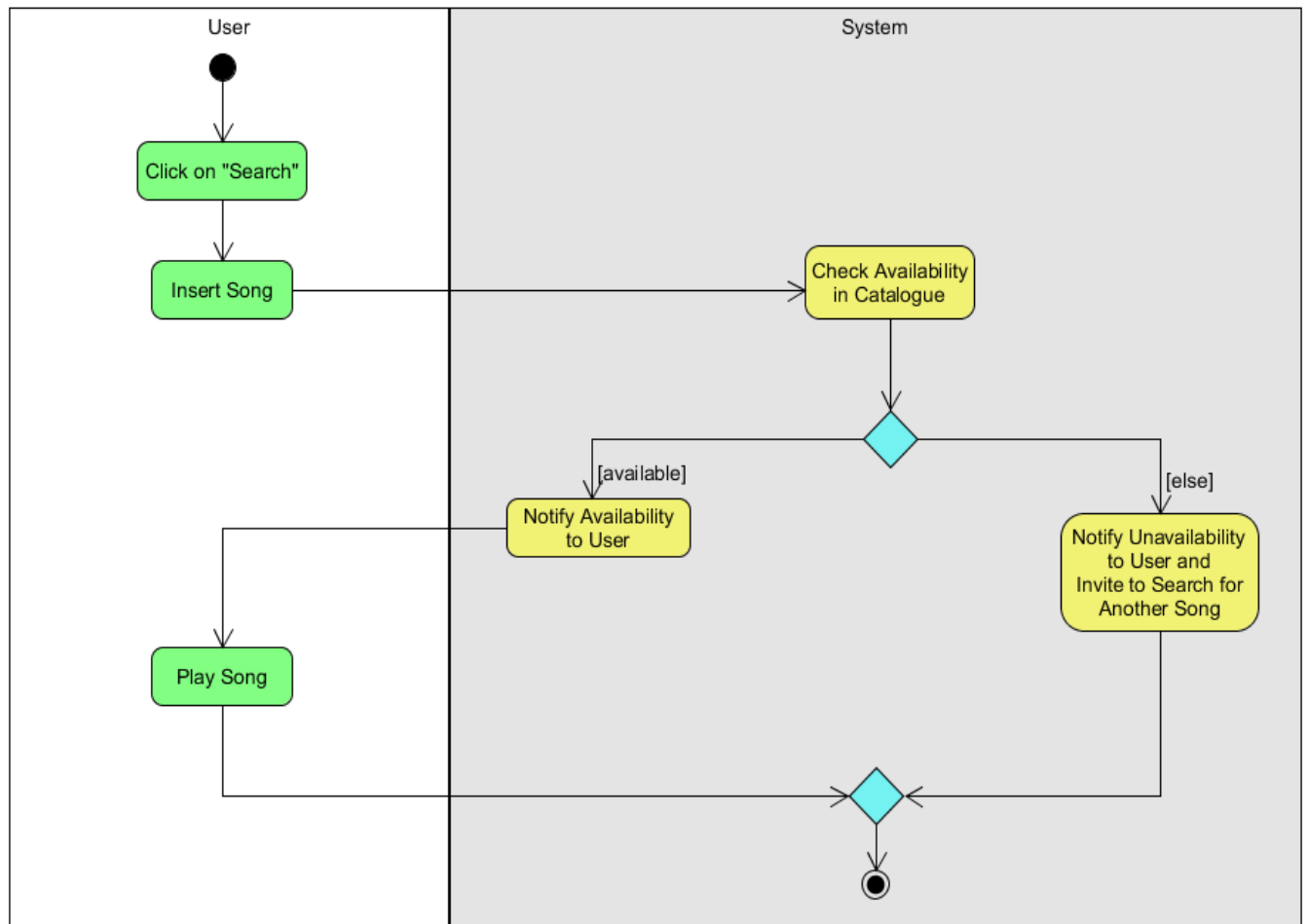
“Listening” use case

This use case is performed by the user. The user is an actor that is pre-registered in the system. There are two types of user: premium user with access to the full catalogue of songs and free user that has a limited access to fewer songs plus adverts. This case allows the user to search and stream a song online if available in the platform.

“Listening” Use Case – Table description

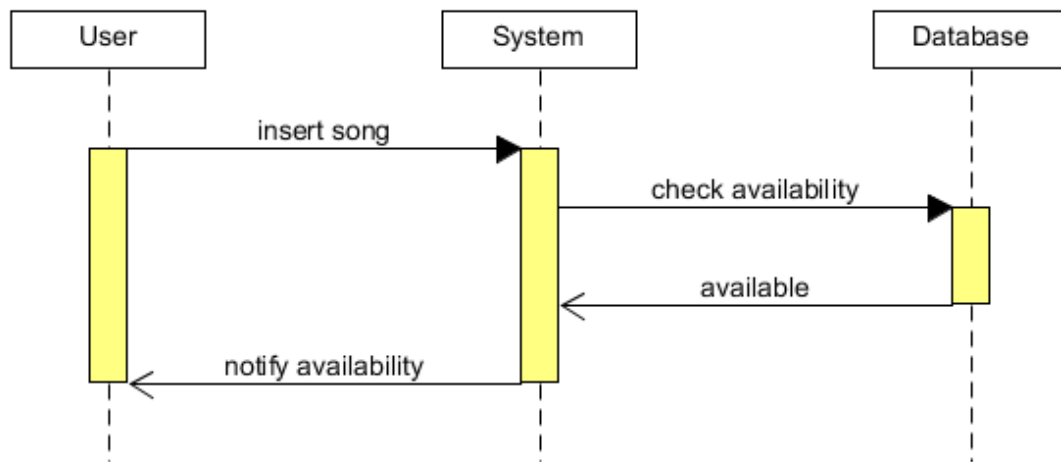
Name	Listening
Description	It enables the user to search for a song and listen to it.
Basic flow	<ol style="list-style-type: none">1. The user clicks on “Search”.2. The user inserts the title of a song and confirms.3. The system checks the availability of the song in the catalogue, contained in the database.4. The system notifies the user that the song is available.5. The user plays and listen to the song.
Alternative flows	<ul style="list-style-type: none">• At step 3 of the basic flow: Not available<ol style="list-style-type: none">1. The system notifies the user that the song is not available and invites them to search for another song.
Pre-conditions	The user has to log in. App downloaded on a smartphone, tablet, or computer.
Post-conditions	Success: the user listens to the song searched. Failure: the user will not listen to the song searched.
Special requirements	Internet connection.

“Listening” Use Case – Activity Diagram



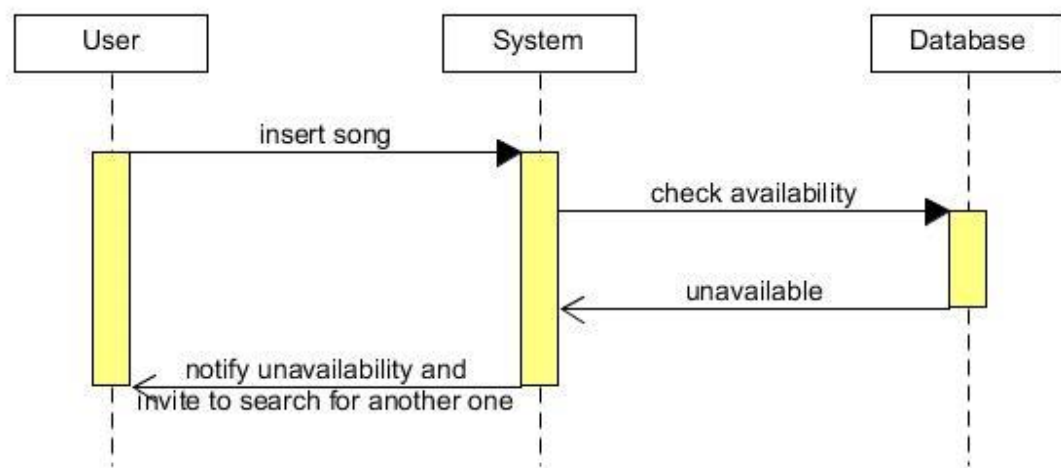
Note: “Invite to Search for Another Song” means that a notification invites the user to start over the process, that is clicking again on “Search”.

“Listening” Use Case – Sequence Diagram (Basic Flow). The user plays and listen to the song.



“Listening” Use Case – Sequence Diagram (Alternative Flow 1).

The system notifies the user that the song is not available and invites them to search for another song.



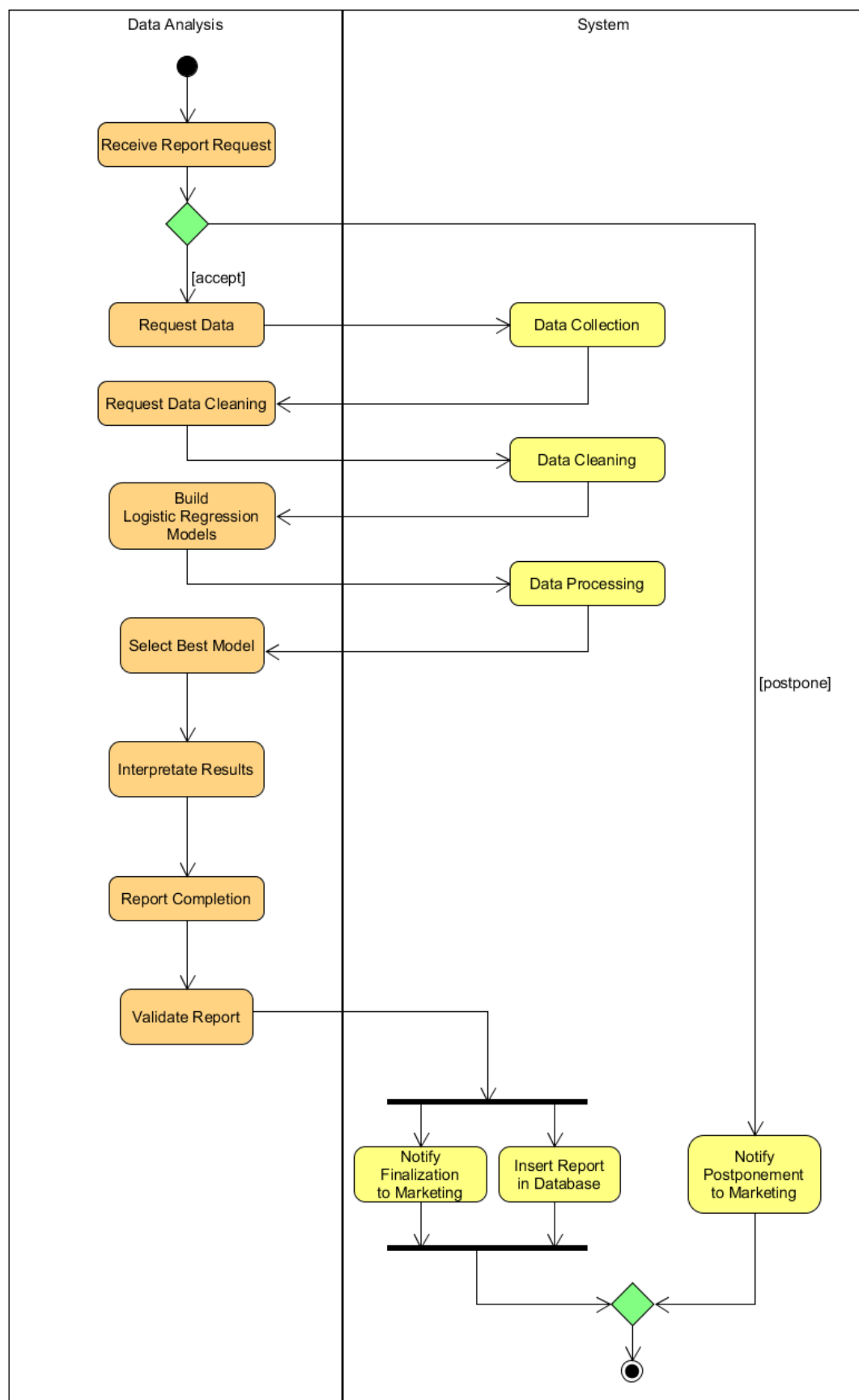
“Churn Report” use case

This use case is performed by the data analyst who collects, cleans and processes data in order to complete a churn analysis report to be sent to the Marketing Department. Examples of useful data for the analysis are users’ sex and nationality, and the frequency of users’ premium subscriptions.

“Churn Report” Use Case – Table description

Name	Churn Report
Description	It helps the Data Analysis Department to prepare a churn analysis report that will be useful for strategic decision-making aimed to prevent churns and enhance the user’s loyalty.
Basic flow	<ol style="list-style-type: none">1. The Data Analyst receives by the system a request for a churn analysis report from the Marketing.2. The Data Analyst identifies the useful data needed for the churn analysis and requests them to the system.3. The system collects the selected data from the database.4. The Data Analyst requests a data cleaning to the system.5. The system removes or modifies data that is incorrect, incomplete, irrelevant, duplicated, missing, or improperly formatted.6. The Data Analyst builds logistic regression models.7. The system processes data and gives an output.8. The Data Analyst goes through the output and selects the best model.9. The Data Analyst interprets the results obtained.10. The Data Analyst formalizes these results by completing a report.11. The Data Analyst goes through the report and validates it.12. The system notifies the finalization of the report to the Marketing.13. The system inserts the report in the database.
Alternative flows	<ul style="list-style-type: none">• At step 1 of the basic flow: Request postponement<ol style="list-style-type: none">1. The system notifies the postponement of the request to the Marketing.
Pre-conditions	-
Post-conditions	Success: The Marketing Department receives the report and analyzes it. Failure: the report has been postponed.
Special requirements	-

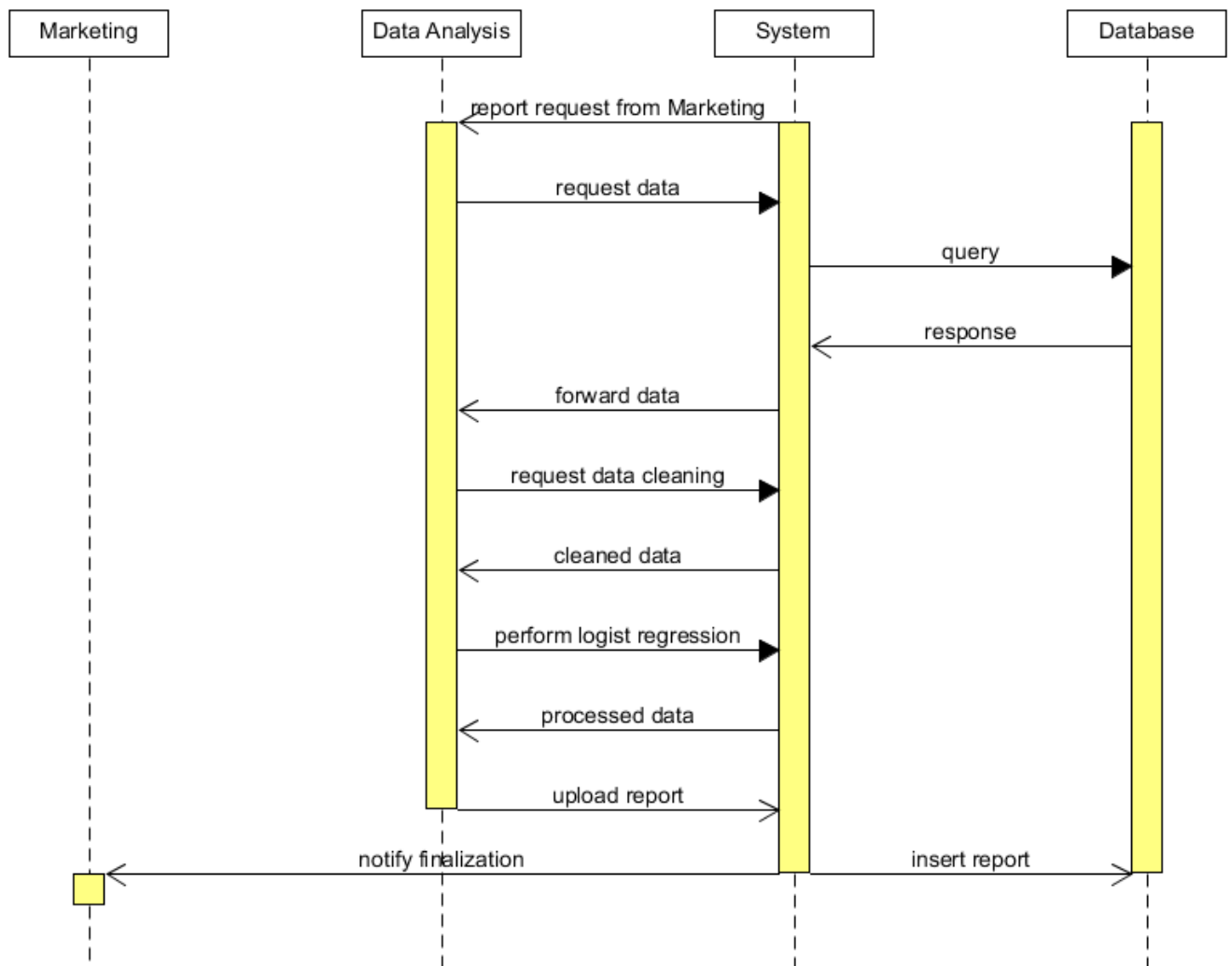
“Churn Report” Use Case – Activity Diagram



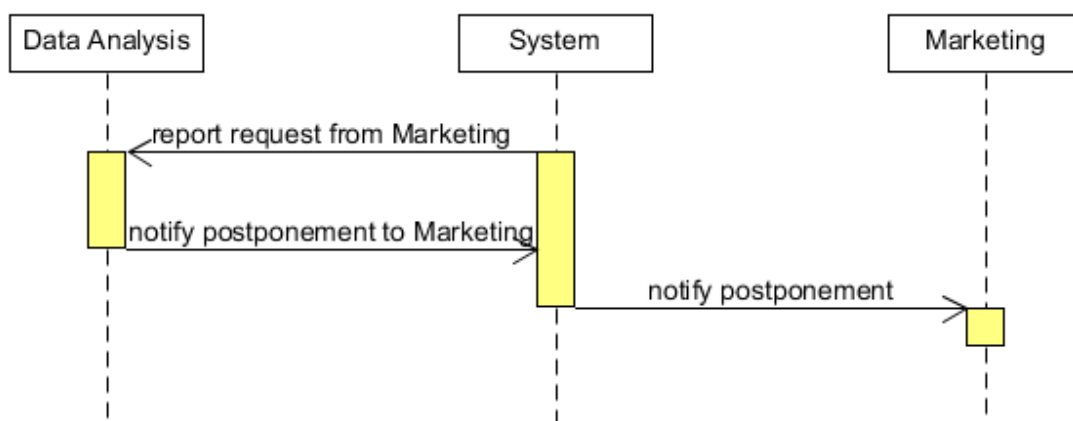
Note: it is implied that the Marketing Department sends a report request to the system which will then notify the Data Analysis about it.

“Churn Report” Use Case – Sequence Diagram (Basic Flow)

Churn report has been completed and added in the database.



“Churn Report” Use Case –Sequence Diagram (Alternative Flow 1). Report completing postponed.



“Bug Fixing” use case

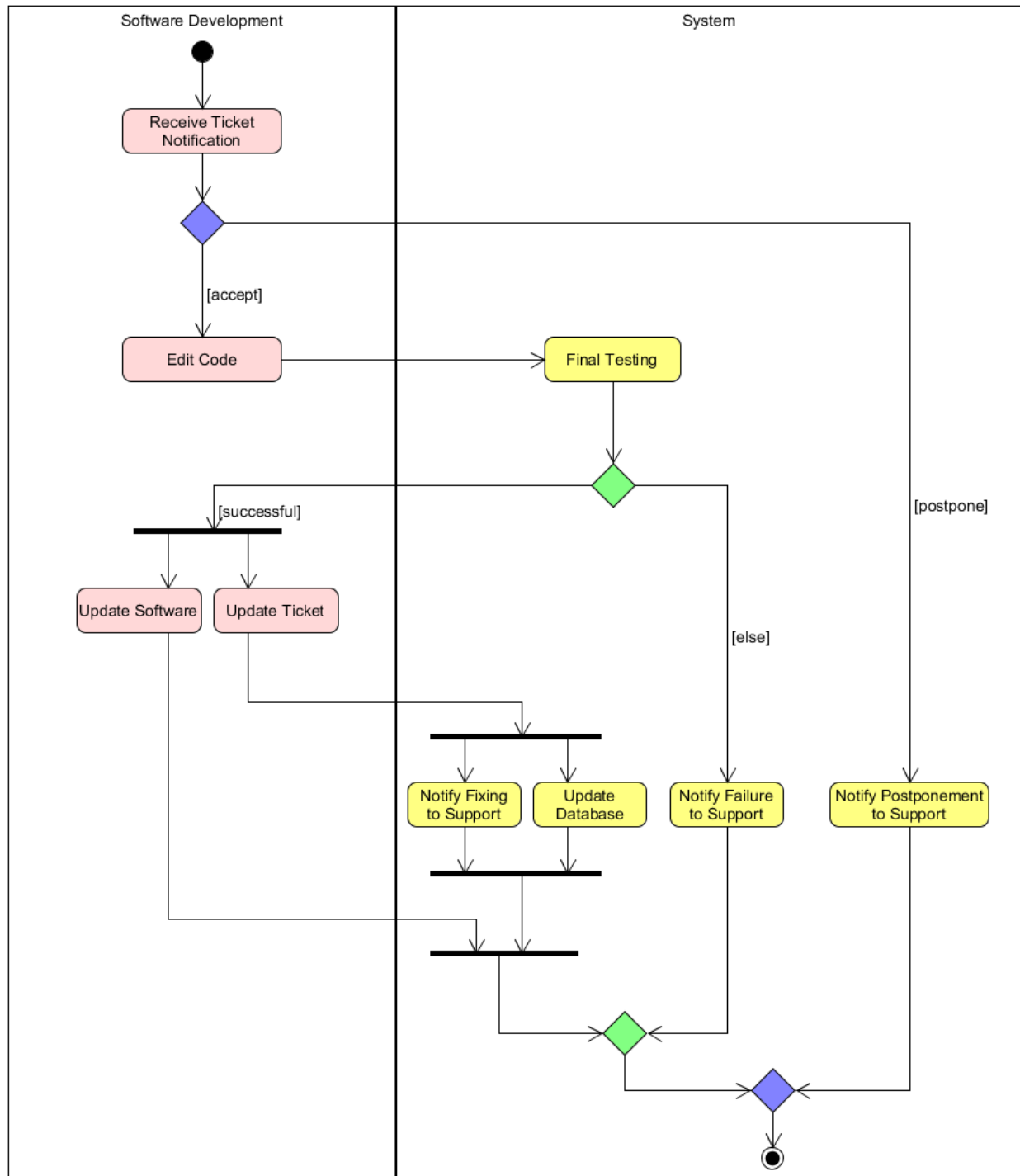
This use case handles the coding part of the bug fixing where the actor, who is the software developer, receives the ticket of the bug and fixes it.

This use case is to make sure that a bug is fixed efficiently and that the user has no issues while using the platform.

“Bug Fixing” Use Case – Table description

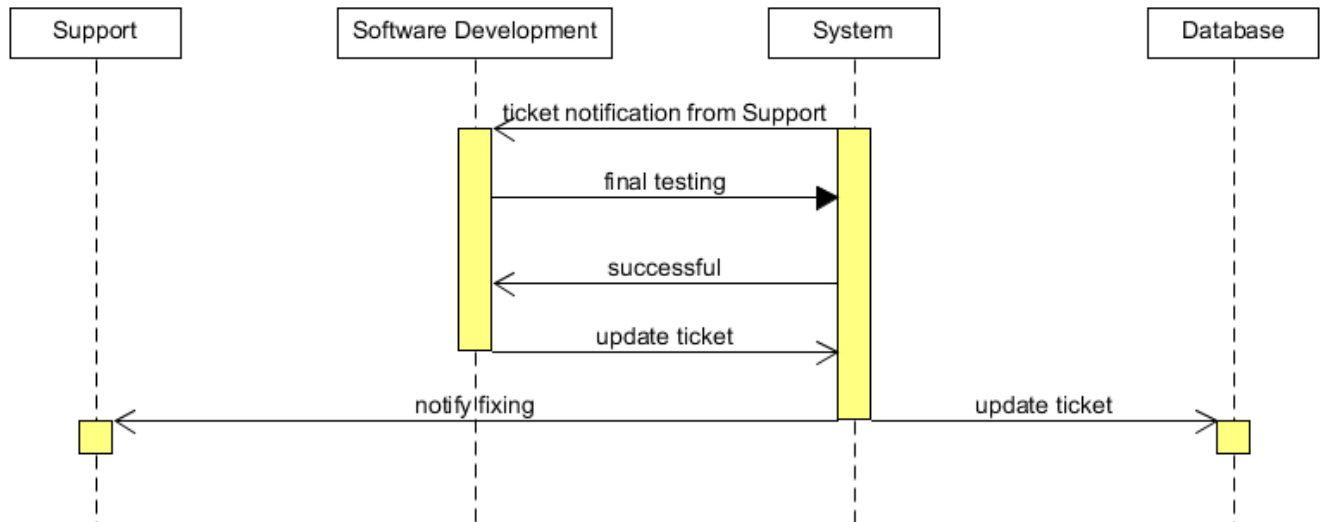
Name	Bug Fixing
Description	It helps the Software Development Department to fix the bug reported by the Support.
Basic flow	<ol style="list-style-type: none">1. The Software Development receives through the system a notification of a ticket creation/updating from the Support.2. The Software Development edits the code and tests it through the system.3. The Software Development requests a final test to the system.4. The Software Development updates the software.5. The Software Development updates the ticket; next, the system notifies the fixing to the Support and updates the database.
Alternative flows	<ul style="list-style-type: none">• At step 3 of the basic flow: Failed Test<ol style="list-style-type: none">1. The system notifies the failure to the Support.• At step 1 of the basic flow: Bug fixing postponement<ol style="list-style-type: none">1. The system notifies to the Support the postponement of the bug fixing request.
Pre-conditions	-
Post-conditions	Success: the bug is fixed, and the user will enjoy a better experience on the app. Failure: bug on going.
Special requirements	-

“Bug Fixing” Use Case – Activity Diagram



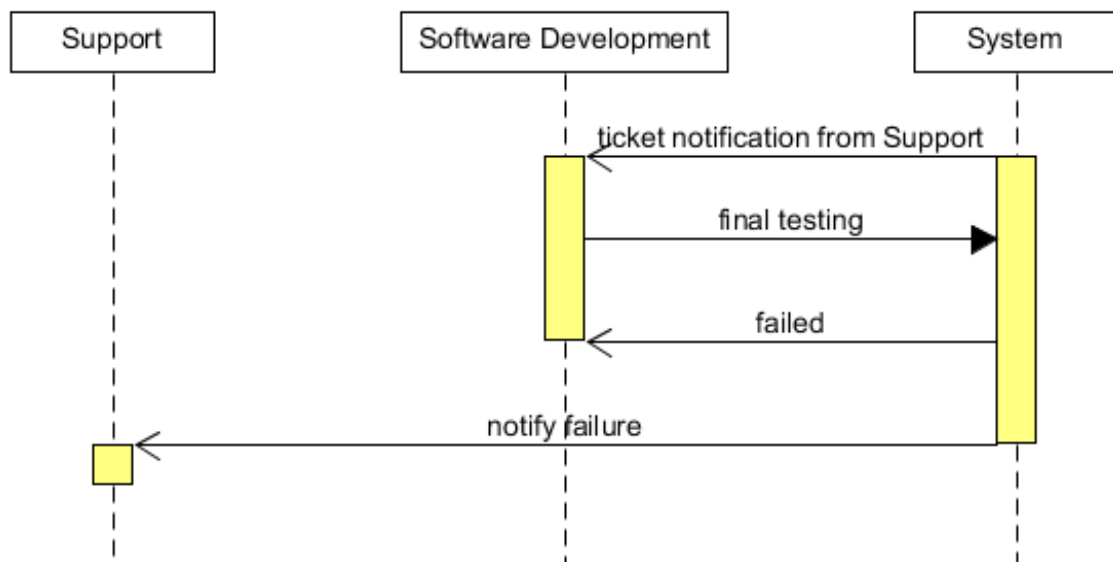
“Bug Fixing” Use Case – Sequence Diagram (Basic Flow).

The Software Development updates the ticket; next, the system notifies the fixing to the Support and updates the database.



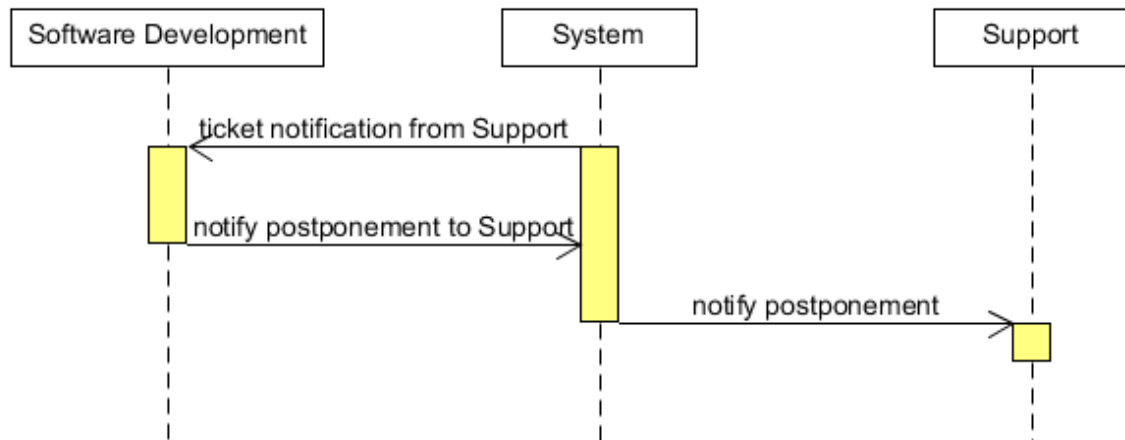
“Bug Fixing” Use Case – Sequence Diagram (Alternative Flow 1).

Failed Test: The system notifies the failure to the Support.



“Bug Fixing” Use Case – Sequence Diagram (Alternative Flow 2)

The system notifies to the Support the postponement of the bug fixing request.



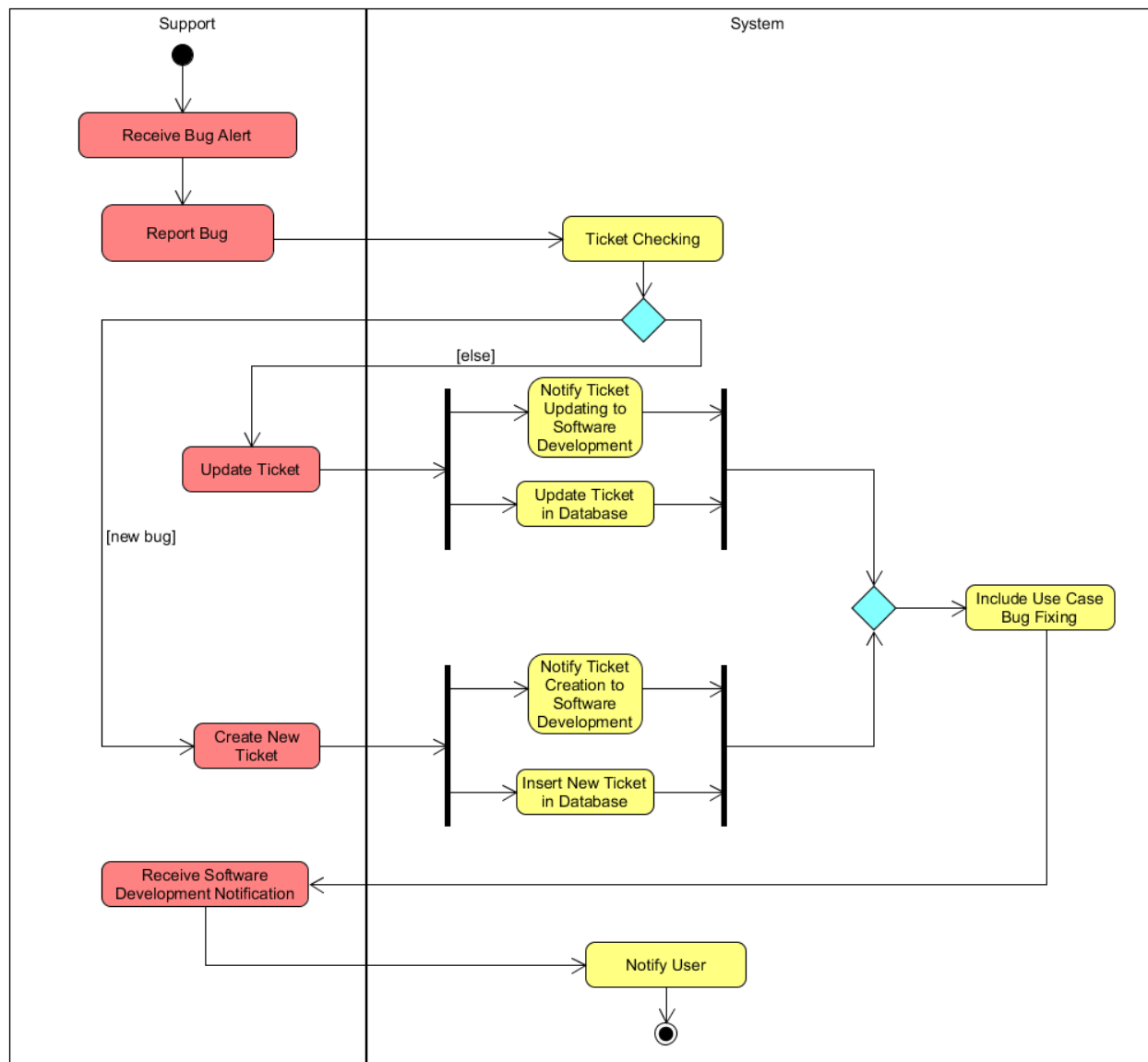
“Bug Management” use case

This is a use case that is meant to handle in the most efficient way the bugs that are faced by the user in the application. The actor of this case is the support assistant who is responsible for receiving and formalizing the bugs reported by the user. After that, the support assistant either creates a new ticket or update an existing one to be processed by the Development Department.

“Bug Management” Use Case – Table description

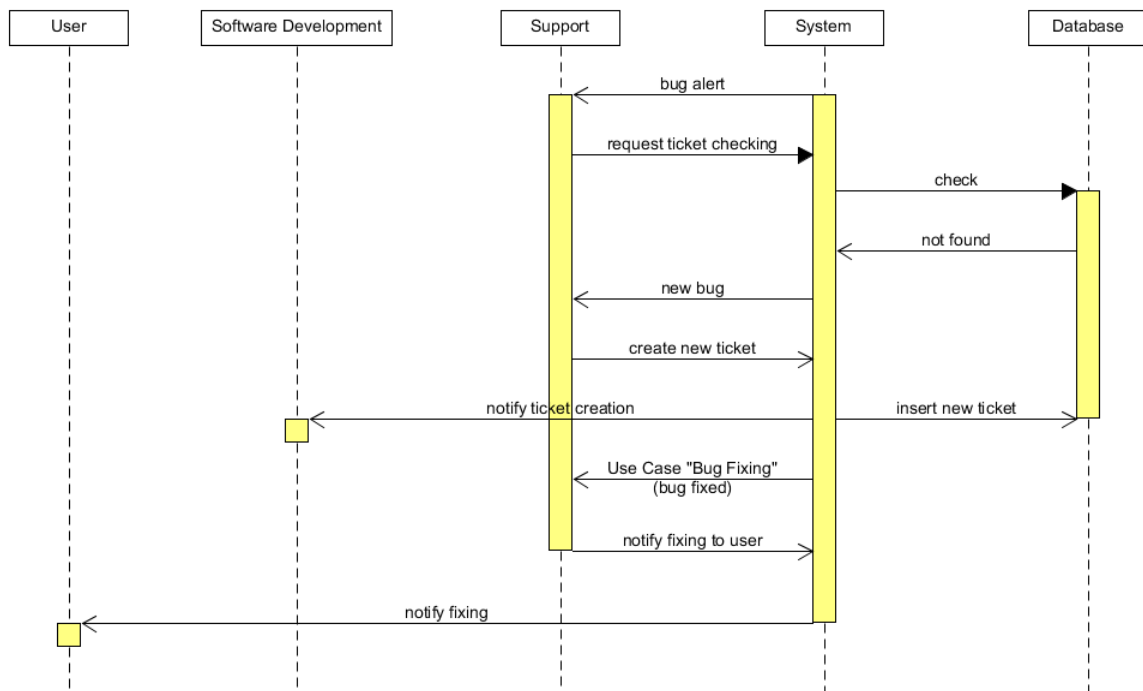
Name	Bug Management
Description	It allows the Support Department to manage bugs by formalizing all the users’ feedback and complaints in order to be comprehensible for the Software Development Department.
Basic flow	<ol style="list-style-type: none">1. The Support receives through the system a bug alert from the user.2. The Support analyzes and reports the bug.3. The system checks if the complaint is related to a new bug (never experienced before).4. The Support creates a new ticket.5. The system notifies the creation of a new ticket to the Software Development.6. The system inserts the new ticket in the database.7. <Include Use Case Bug Fixing>.8. The Support receives a notification about the status of the bug from the Software Development.9. The system notifies the user that the bug has been fixed.
Alternative flows	<ul style="list-style-type: none">• At step 3 of the basic flow: Existing ticket (bug already experienced)<ol style="list-style-type: none">1. The Support updates the ticket related to that specific bug.2. The system notifies the ticket updating to the Software Development.3. The system updates the ticket, relating to that specific bug, in the database.4. Go to step 7 of the basic flow.
Pre-conditions	
Post-conditions	Success: The Support successfully processes the user’s request related to the bug. Failure: none, because all of the users warns are processed and all of the bugs will be fixed sooner or later.
Special requirements	-

“Bug Management” Use Case – Activity Diagram

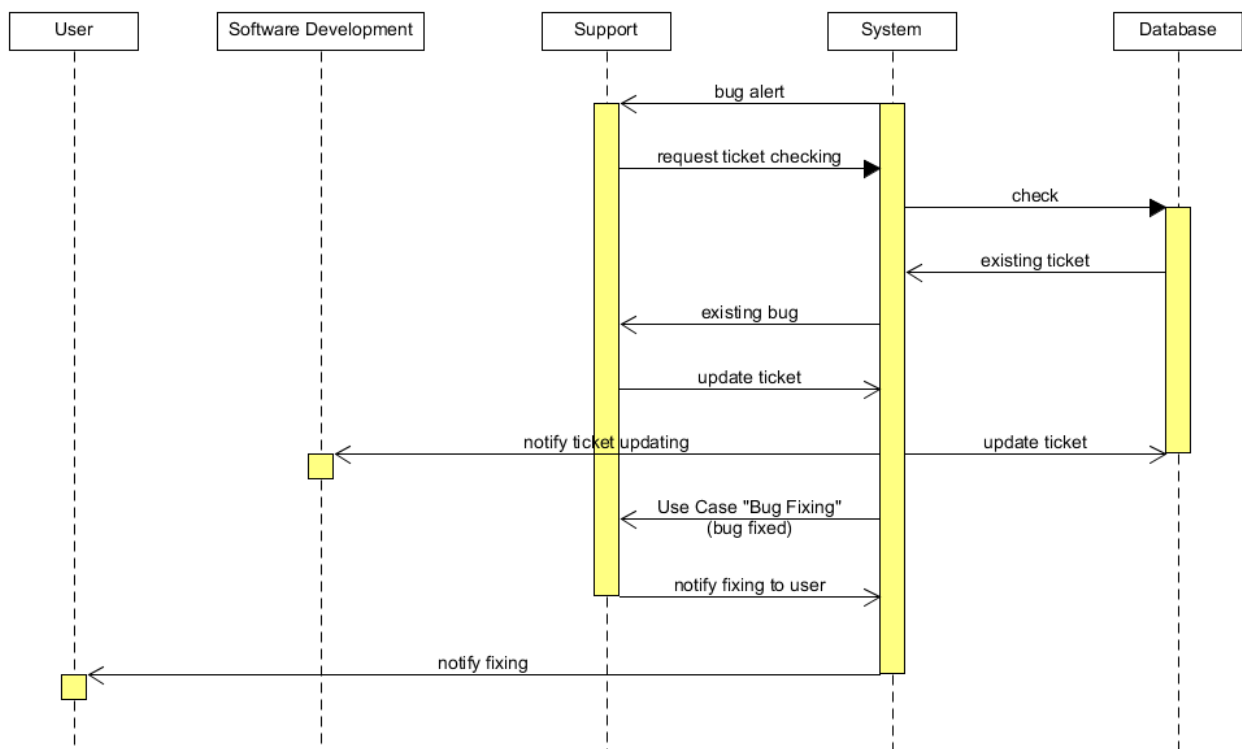


Note: “Report Bug” refers to the Support analyzing and formalizing the user’s bug warning.

“Bug Management” Use Case – Sequence Diagram (Basic Flow) – with basic flow of “Bug Fixing” Use Case. Bug successfully managed and fixed.



“Bug Management” Use Case – Sequence Diagram (Alternative Flow 1) – with basic flow of “Bug Fixing” Use Case. Bug already experienced.



2 Design

2.1 Database Design

In order to support the functionalities required by Omegastream the database has to store information about users, subscriptions, emails, songs, albums, artists, streams, tickets, bugs reported, updates developers, support assistants, data analysts and churn reports.

For the use cases “Create Account” and “Login”, the following data is needed:

- For users: username, password, email, name, surname, sex, date of birth, nationality;
- For subscriptions: code, type (which can be either premium or free), date of beginning, date of end, frequency (requested when subscribing a premium account, meaning it takes a value if the attribute “type” is premium; it can be monthly, quarterly, biannual, or annual);
- For emails: code, subject, text, date.

Regarding the use case “Listening”:

- For streams: date, time, device (computer, smartphone, or tablet);
- For songs: code, title, genre, year, length, type (which can be either free or premium and it takes always a value in order to make possible the catalogue filtering);
- For artists: code, name, date of birth, nationality;
- For albums: code, title, date of release.

With respect to the use cases “Bug Management” and “Bug Fixing”, the Omegastream’s DB will have to store the data listed below:

- For bugs reported: code, info, date, time;
- For support assistants: code, email, password, name, surname;
- For tickets: code, description, date of creation, time of creation;
- For updates: date, info (regarding the fixing, the failure, or the postponement), time;
- For developers: code, email, password, name, surname.

Most of the data mentioned above might be useful covariates to build a model from, however it is important to store specific data about the reports. As concerning the use case “Churn Report”:

- For churn reports: code, title, date, time;
- For data analysts: code, email, password, name, surname.

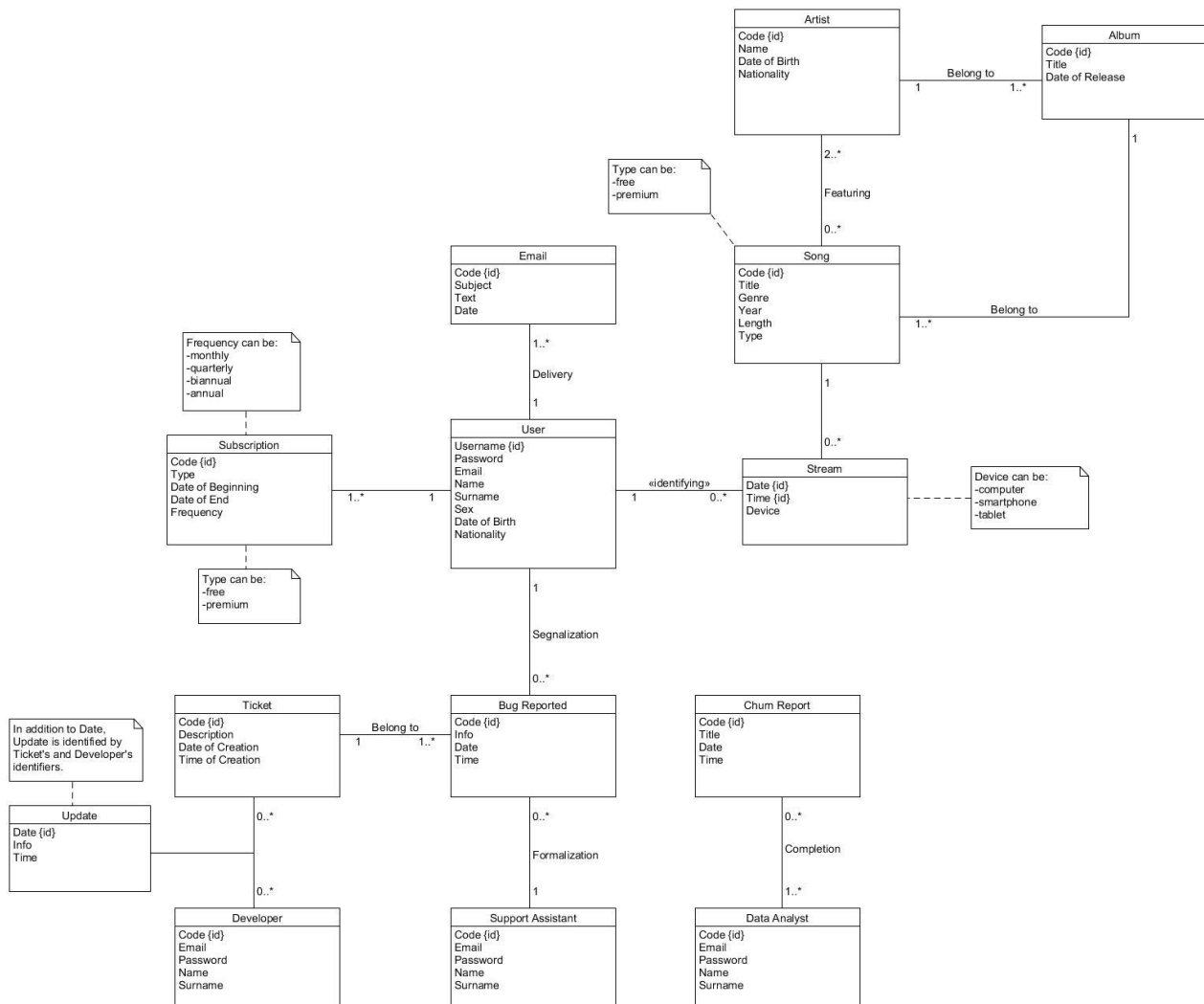
Developers, support assistants and data analysts have their own email and password in order to login to the system. Their login is different from the user’s one and for the moment it has not been modeled. It will be probably taken into account in future developments of the information system with regards to the internal needs of Omegastream.

Moreover, the catalogue is composed of albums which can be either proper ones (with several songs in it) or made up of just a single song. This is aimed to model the real circumstance in which an artist releases just a single song; in order for that to be available on the information system, it has to refer to an album, meaning an album containing only that specific song is added in the DB.

This is because an artist exists in the DB only if they have at least one album.

2.1.1 Conceptual Model

The conceptual model, made by capturing the main ideas behind the use cases analysed, is represented by the following class diagram containing 14 classes and the relationships among them:



- A user is allowed to complete at least one at most n subscriptions, while a specific subscription can be linked to only one user;
- A user can receive at least one (premium subscription discount after registration) at most n emails, while an email is always referred to a specific user;
- A user can stream songs min 0 max n times and a specific stream is always referred to one particular user;
- A song belongs to one and only one album, while an album can be made up of min one max n songs;
- An album belongs to only one artist, while an artist can be linked to at least one at most n albums;
- An artist can feature in min 0 max n songs, if the song is a featuring is linked to at least two at most n artists;

- A user can report min 0 max n bugs and a specific bug warning is reported by one user;
- A bug (reported) is formalized by one particular support assistant; a support assistant can formalize min 0 max n bug (reported);
- A bug reported is referred to one ticket and a ticket is referred to at least one at most n bug (reported);
- A ticket can be updated by min 0 max n developers, while a developer updates min 0 max n tickets.

2.1.2 Relational Model

User (Username, Password, Email, Name, Surname, Sex, DateofBirth, Nationality)

- Username VARCHAR(15), Primary Key
- Password VARCHAR(20)
- Email VARCHAR(45)
- Name VARCHAR(45)
- Surname VARCHAR(45)
- Sex VARCHAR(1), it can be either “M” for male or “F” for female
- DateofBirth DATE
- Nationality VARCHAR(30)

Subscription (Code, Type, DateofBeginning, DateofEnd, Type, User)

- Code INT(6), Primary Key
- Type VARCHAR(1), it can be either “F” for free or “P” for premium
- DateofBeginning DATE
- DateofEnd DATE
- Frequency VARCHAR(20), it can be “monthly”, “quarterly”, “biannual”, or “annual”; it has a value only if Type=“P”
- User VARCHAR(15), NOT NULL

R.I.C.: Subscription [User] \sqsubseteq_{FK} User [Username]

Email (Code, Subject, Text, Date, User)

- Code INT(6), Primary Key
- Subject VARCHAR(50)
- Text VARCHAR(1000)
- Date DATE
- User VARCHAR(15), NOT NULL

R.I.C.: Email [User] \sqsubseteq_{FK} User [Code]

Artist (Code, Name, DateofBirth, Nationality)

- Code INT(6), Primary Key
- Name VARCHAR(45)
- DateofBirth DATE
- Nationality VARCHAR(30)

Album (Code, Title, DateofRelease, Artist)

- Code INT(6), Primary Key
- Title VARCHAR(50)
- DateofRelease DATE
- Artist INT(6), NOT NULL

R.I.C.: Album [Artist] \sqsubseteq_{FK} Artist [Code]

Song (Code, Title, Genre, Year, Length, Type, Album)

- Code INT(6), Primary Key

- Title VARCHAR(50)
- Genre VARCHAR(30)
- Year INT(4)
- Length TIME
- Type VARCHAR(1), NOT NULL, it can be either “F” for free or “P” for premium
- Album INT(6), NOT NULL

R.I.C.: Song [Album] \sqsubseteq_{FK} Album [Code]

Stream (User, Date, Time, Song, Device)

- User VARCHAR(15), Primary Key
- Date DATE, Primary Key
- Time TIME, Primary Key
- Song INT(6), NOT NULL
- Device VARCHAR(10), it can be “computer”, “smartphone”, or “tablet”

R.I.C.: Stream [User] \sqsubseteq_{FK} User [Username],
Stream [Song] \sqsubseteq_{FK} Song [Code]

Featuring (Song, Artist)

- Song INT(6), Primary Key
- Artist INT(6), Primary Key

R.I.C.: Featuring [Song] \sqsubseteq_{FK} Song [Code],
Featuring [Artist] \sqsubseteq_{FK} Artist [Code]

SupportAssistant (Code, Email, Password, Name, Surname)

- Code INT(6), Primary Key
- Email VARCHAR(45)
- Password VARCHAR(20)
- Name VARCHAR(45)
- Surname VARCHAR(45)

Ticket (Code, Description, DateofCreation, TimeofCreation)

- Code INT(6), Primary Key
- Description VARCHAR(300)
- DateofCreation DATE
- TimeofCreation TIME

BugReported (Code, Info, Date, Time, User, SupportAssistant, Ticket)

- Code INT(6), Primary Key
- Info VARCHAR(300)
- Date DATE
- Time TIME
- User VARCHAR(15), NOT NULL
- SupportAssistant INT(6), NOT NULL
- Ticket INT(6), NOT NULL

R.I.C.: BugReported [User] \sqsubseteq_{FK} User [Username],

BugReported [SupportAssistant] \sqsubseteq_{FK} SupportAssistant [Code],
BugReported [Ticket] \sqsubseteq_{FK} Ticket [Code]

Developer (Code, Email, Password, Name, Surname)

- Code INT(6), Primary Key
- Email VARCHAR(45)
- Password VARCHAR(20)
- Name VARCHAR(45)
- Surname VARCHAR(45)

Update (Developer, Ticket, Date, Info, Time)

- Developer INT(6), Primary Key
- Ticket INT(6), Primary Key
- Date DATE, Primary Key
- Info VARCHAR(300)
- Time TIME

R.I.C.: BugFixed [Developer] \sqsubseteq_{FK} Developer [Code],
BugFixed [Ticket] \sqsubseteq_{FK} Ticket [Code]

DataAnalyst (Code, Email, Password, Name, Surname)

- Code INT(6), Primary Key
- Email VARCHAR(45)
- Password VARCHAR(20)
- Name VARCHAR(45)
- Surname VARCHAR(45)

ChurnReport (Code, Title, Date, Time)

- Code INT(6), Primary Key
- Title VARCHAR(80)
- Date DATE
- Time TIME

Completion (DataAnalyst, ChurnReport)

- DataAnalyst INT(6), Primary Key
- ChurnReport INT(6), Primary Key

R.I.C.: Completion [DataAnalyst] \sqsubseteq_{FK} DataAnalyst [Code],
Completion [ChurnReport] \sqsubseteq_{FK} ChurnReport [Code]

Note: R.I.C. stands for Referential Integrity Constraint.

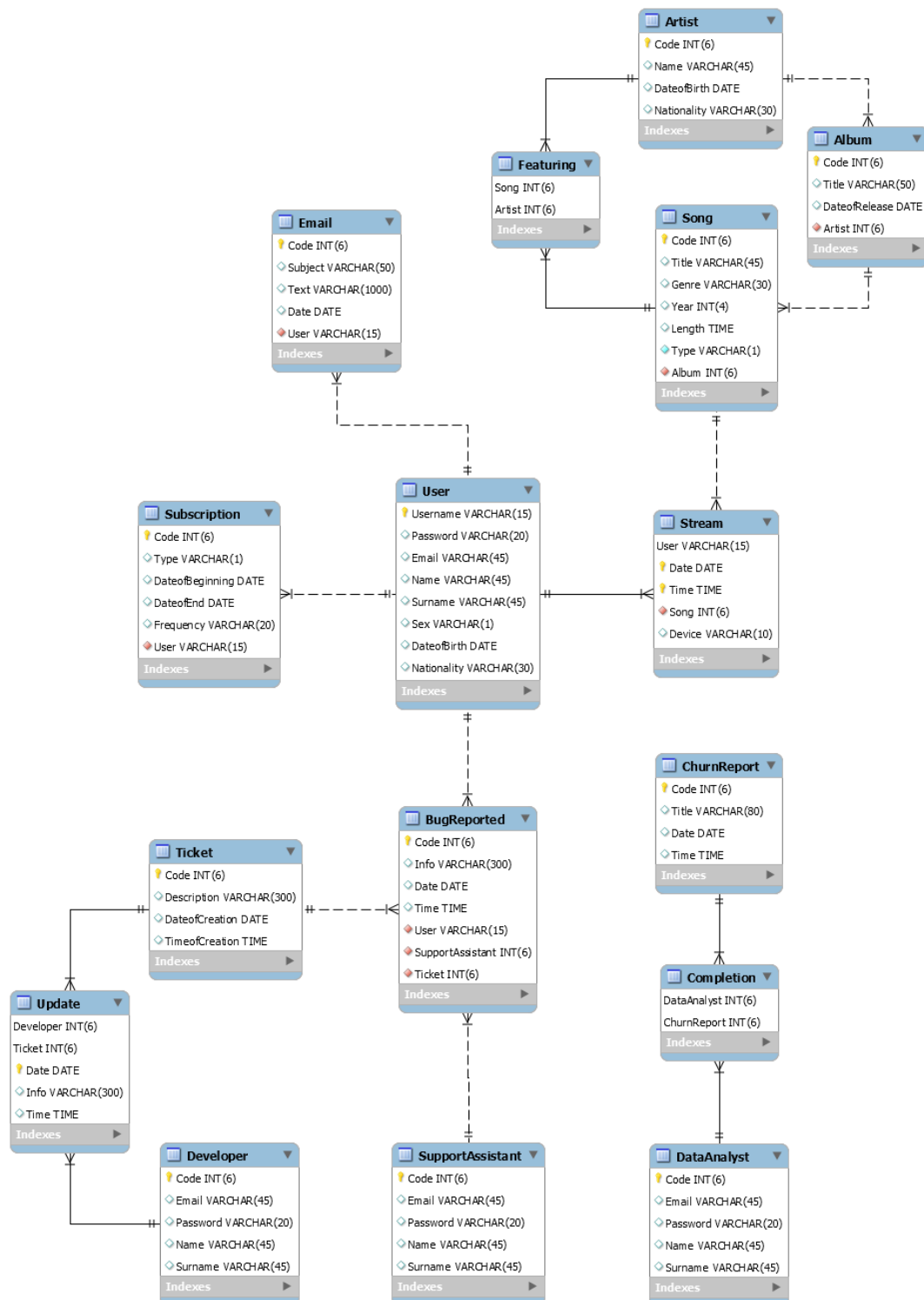
Further constraints:

- In the relational model all the multiplicity constraints 1..* have been relaxed and turned into the 0..* type. This is because the first type is not supported in the relational model. For example, even if an artist is always linked to at least one album, in the relational model obtained this is not necessarily true.
The application logic layer of the information system is in charge of the maintenances of such constraints; for example, the application will have to add a tuple referred to an album every time a new artist is added in the database;
- The application is in charge of the maintenance of the multiplicity constraint 2..* that links one song to two or more artists through the relationship “Featuring”.

To make the implementation process smoother, it has been decided to express each attribute as a single word, e.g. “Date of Birth” as “DateofBirth”.

3 Implementation

The database supporting the Information System has been implemented on MySQL. In order to create a graphic model, based on the relational one, it has been used MySQL Workbench and, to quickly convert the graphic model into a physic database which exists on a target MySQL server, it has been used the forward engineering functionality provided by such DBMS.



Appendix

Below there are examples of queries potentially useful to manage the processes analysed:

- In order for the Data Analysis Department to perform an insightful churn analysis, it will be necessary to collect key information regarding the users, such as sex, nationality and frequency of premium subscriptions

```
1 SELECT user.Username, user.Sex, user.Nationality, subscription.Frequency
2 FROM progetto.subscription, progetto.user
3 WHERE user.Username=subscription.User
4 AND subscription.Type='P';
```

Username	Sex	Nationality	Frequency
mariorossi5	M	Italian	monthly
luisaliverdi	F	Italian	annual

- Music genres streamed by millennials (users aged 25 or less)

```
1 SELECT DISTINCT song.Genre
2 FROM progetto.song, progetto.stream, progetto.user
3 WHERE song.Code=stream.Song
4 AND user.Username=stream.User
5 AND user.DateofBirth >= 1995-01-01;
```

Genre
R&B

- Number of bugs reported in the first quarter of 2020

```
1 SELECT COUNT(*) AS Number_of_Bug_Reported_in_the_first_quarter
2 FROM progetto.bugreported
3 WHERE bugreported.Date >= '2020-01-01'
4 AND bugreported.Date <= '2020-03-31';
```




























Number_of_Bug_Reported_in_the_first_quarter
2

Glossary of Terms

Term	Description	Synonyms	Linked Terms
User	A person who has signed up on the platform and uses it.	Subscriber, Customer, Client, End-user	-
Visitor	Someone who simply views/goes to Omegastream website. They have no account and are likely to create one.	-	-
Churn Analysis	Evaluation of the company's customer loss rate in order to reduce it.	-	Report
Bug	An error, flaw or fault in the software that causes it to produce an incorrect or inspected result, or to behave in unintended ways.	Error	Complaint, Bug Management, Bug Fixing, Support Assistant, Developer, Support Section, Ticket
Support Section	A dedicated part of the user interface where it is possible to report bugs.	-	Bug Management, Bug Fixing, Support Assistant, Ticket, User
Customer Experience	The product of an interaction between the organization and a customer over the duration of their relationship.	User Experience	Loyalty to the platform
Sign Up	To register on the platform and create an account.	-	-
Model Estimation	Detecting the covariates/factors that most significantly affect the response variable.	-	Logistic Regression
Logistic Regression	It is used to model the probability of a certain class or event existing such as pass/fail, win/lose etc..	-	-
Ticket	Formalization of a bug the end-user might have experienced. Its purpose is to formalize the issue reported by	-	-

	the user so that it can be easily understood by the Software Development. It is always created by a support assistant and it can be updated by both developers and support assistants.		
Data Cleaning	Process of detecting and correcting corrupt or inaccurate data from a dataset.	-	Churn Analysis, Data Collection, Data Process
Data Collection	Process of gathering and measuring information on targeted variables in an established system.	-	Churn Analysis, Data Cleaning, Data Process
Data Process	Collection and manipulation of items of data that produce meaningful information.	-	Churn Analysis, Data Collection, Data Cleaning

Attached files

-  1_activitydiagram_Process1.uxf
-  1_activitydiagram_Process2.uxf
-  1_activitydiagram_Process3.uxf
-  2_usecasediagram.uxf
-  3_activitydiagram_BUG FIXING.uxf
-  3_activitydiagram_BUG MANAGEMENT.uxf
-  3_activitydiagram_CHURN REPORT.uxf
-  3_activitydiagram_CREATE ACCOUNT.uxf
-  3_activitydiagram_LISTENING.uxf
-  3_activitydiagram_LOGIN.uxf
-  4_sequencediagram1_BUG FIXING.uxf
-  4_sequencediagram1_BUG MANAGEMENT.uxf
-  4_sequencediagram1_CHURN REPORT.uxf
-  4_sequencediagram1_CREATE ACCOUNT.uxf
-  4_sequencediagram1_LISTENING.uxf
-  4_sequencediagram1_LOGIN.uxf
-  4_sequencediagram2_BUG FIXING.uxf
-  4_sequencediagram2_BUG MANAGEMENT.uxf
-  4_sequencediagram2_CHURN REPORT.uxf
-  4_sequencediagram2_CREATE ACCOUNT.uxf
-  4_sequencediagram2_LISTENING.uxf
-  4_sequencediagram2_LOGIN.uxf
-  4_sequencediagram3_BUG FIXING.uxf
-  4_sequencediagram3_LOGIN.uxf
-  5_classdiagram.uxf
-  6_graphicmodelMysqlWorkbench.mwb
-  7_database.sql