
Software Requirements Specification

for

Music Player(NAAS)

Version 1.0

Prepared by-

Nishtha Varshney PES1201801229

Kara Akhila PES120180311

Maknoor Shalini PES1201800253

Animeha Bakshi PES1201801661

PES UNIVERSITY

6.02.2021

Table of Contents

Table of Contents	ii
Revision History	ii
1. Introduction	3
1.1 Purpose	3
1.2 Intended Audience and Reading Suggestions	3
1.3 Product Scope	3
1.4 References	4
2. Overall Description	4
2.1 Product Perspective	4
2.2 Product Functions	4
2.3 User Classes and Characteristics	5
2.4 Operating Environment	6
2.5 Design and Implementation Constraints	6
2.6 Assumptions and Dependencies	6
3. External Interface Requirements	7
3.1 User Interfaces	7
3.2 Software Interfaces	8
3.3 Communications Interfaces	8
4. Analysis Model	8
4.1 Use Case Diagram	9
4.2 ER Diagram	10
5. System Features	10
5.1 Authentication	10
5.2 Search songs	11
5.3 Managing playlist	12
5.4 User Ratings	12
5.5 Play/pause/next/prev	13
5.6 Recommendation	13
6. Other Nonfunctional Requirements	14
6.1 Performance Requirements	14
6.2 Safety Requirements	14
6.3 Security Requirements	14
6.4 Software Quality Attributes	14
6.5 Business Rules	14
7. Other Requirements	15
Appendix A: Glossary	15
Appendix B: Field Layouts	15
Appendix C: Requirement Traceability matrix	16

1. Introduction

1.1 Purpose

This Software requirement specification(SRS) document gives a detailed description of a music streaming application 'NAAS' version 1.0. The document intends to explain various features, interfaces and system constraints for the application. It establishes an overview of the interaction between the application and the end user, summary-level use cases intended and how the user experience is made better by making the interface convenient considering their preferences from their previous actions. An overview of the server side functionalities and databases has also been discussed in this document. This gives an overall description of the product.

1.2 Intended Audience

The document is primarily intended to developers of this product, for future reference about the project overview and the software users. Brief summaries of what each section in this document provides are given below:

- section 1: Introduction
This section provides a summary of the project including objectives, product scope.
- section 2: Description
The sections briefly describe the product, functionalities and use cases of the product, system requirements and various constraints involved.
- section 3: External interface requirements
Characteristics of various interfaces involved among the various phases of the software product are provided in this section.
- section 4: Analysis model
- section 5: System features
- section 6: Non-functional requirements
- section 7: Other requirements
This field contains any additional information that might be useful for the readers.

1.3 Product Scope

NAAS is a music streaming application which intends to mimic popular applications such as Wynn, spotify. The application contains two components: User side component and a Server side component. User side component comprises a graphical interface which is convenient for the users integrated to the backend. And the server side takes care of user information and preferences using which it also makes recommendations to the users from its database which helps them discover similar music of their interest. Every logged in user is displayed with a set of recommended songs which might interest them. User's profile data and their listening experience data is combinedly used to make suggestions to the user .The user can either choose to play a song from the

recommended list or can proceed to the search interface to make a query for a song. Basic music player functionalities such as play, pause, next options are also available.

1.4 References

- https://github.com/ProjectRecommend/docs/blob/master/design-docs/SRS_final.md
- Design and implementation of the music player based on Android .Junli Xu, College of Information Engineering, Jiangxi University of Technology, Jiangxi Nanchang.
- SRS for project Music Box.net Version 1.7.5.5, Prepared by Katerina Fotiou.

2. Overall Description

2.1 Product Perspective

NAAS is a user-friendly web application that allows users to search and play music of their interests. In addition to searching for the songs this application recommends songs based on the users interest of genre or singers. This system consists of three components packed together

- Music player - for playing music from the library
- Recommender - on the bases of present track playing and the users interests application generate suggestions
- Database - maintains list of tracks along with metadata

2.2 Product Functions

This software lets users play tracks available in the library. Users can also create a playlist where they can add their favourite tracks and while playing music users can get a list of suggested tracks which are most closely related with the present track in terms of their metadata tags and the interest of the user.

Users can perform the following functions:

- Users can login/Sign up
- Users can play/pause/stop/seek
- Users can create a playlist.
- Users can add or remove tracks
- Users can give ratings to the tracks.

System can perform the following functions:

- Checks user login details.
- Show metadata of the tracks.
- Manage the playlist.
- Update the app.
- Recommend songs to users.

2.3 User Classes and Characteristics

The software can be used by the following user categories:

- Users that listen to music. These users could be of any age, with no extraordinary information. Everyone using a computer can use this application.
- Developers of all ages who can comprehend the program's source code and extend or improve it. Should know about the programming language that the product is written in, to understand exactly what it does and how it does it and help in maintaining the basic functionalities of the application.

2.4 Operating Environment

The web application is supported by the following web browsers:

- Chrome
- Firefox
- Edge
- Opera
- Safari

2.5 Design and Implementation Constraints

- Since we need user profile information and their interests while building up the software, to figure out ongoing and adequate information real time data can be a problem for the developer because of regulatory policies.
- For recommendation we are using metadata provided by the users on the basis of their interests to implement content based filtering. Content based filtering generates less accurate results but due to non availability of the user data we cannot create recommendations using collaborative filtering(which uses user behaviour to analyze), which generates more accurate results.

2.6 Assumptions and Dependencies

Assumptions - Users should have access to reliable internet services.

Dependencies - We will be re-using the recommender system already available in the market

3. External Interface Requirements

3.1 User Interfaces

The NAAS user interface allows the user to search for any song they wish to listen to.

The user will be provided with a list of artists and genres during the sign-up procedure through which NAAS will recommend music from artists with similar genres, along with providing you with the music of your chosen artist.

Select your top 10 artists

Skip >

🔍 Search

Button

Button

Button

Button

Button

Button

Button

Button

Button

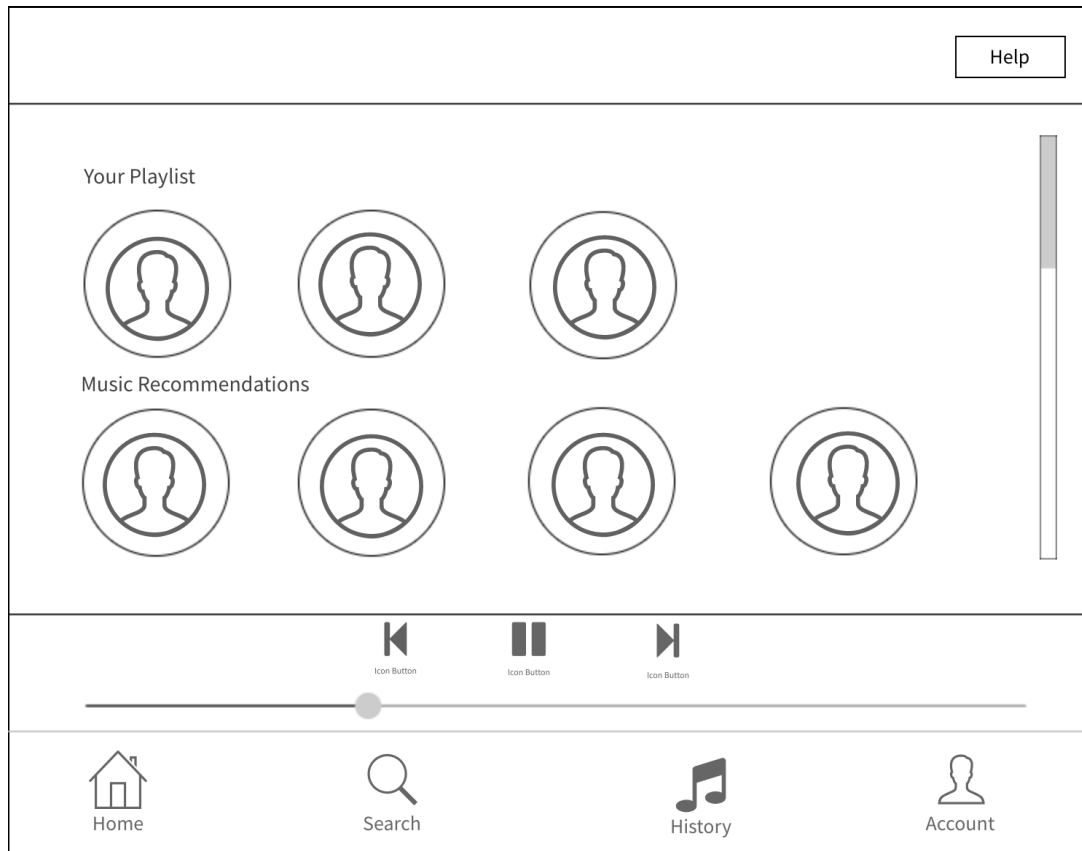
Button

Button

× Button

See more >>

Based on the user's history, NAAS will keep updating the albums that appear in their recommendations.



3.2 Software Interfaces

We have the client side and the server side.

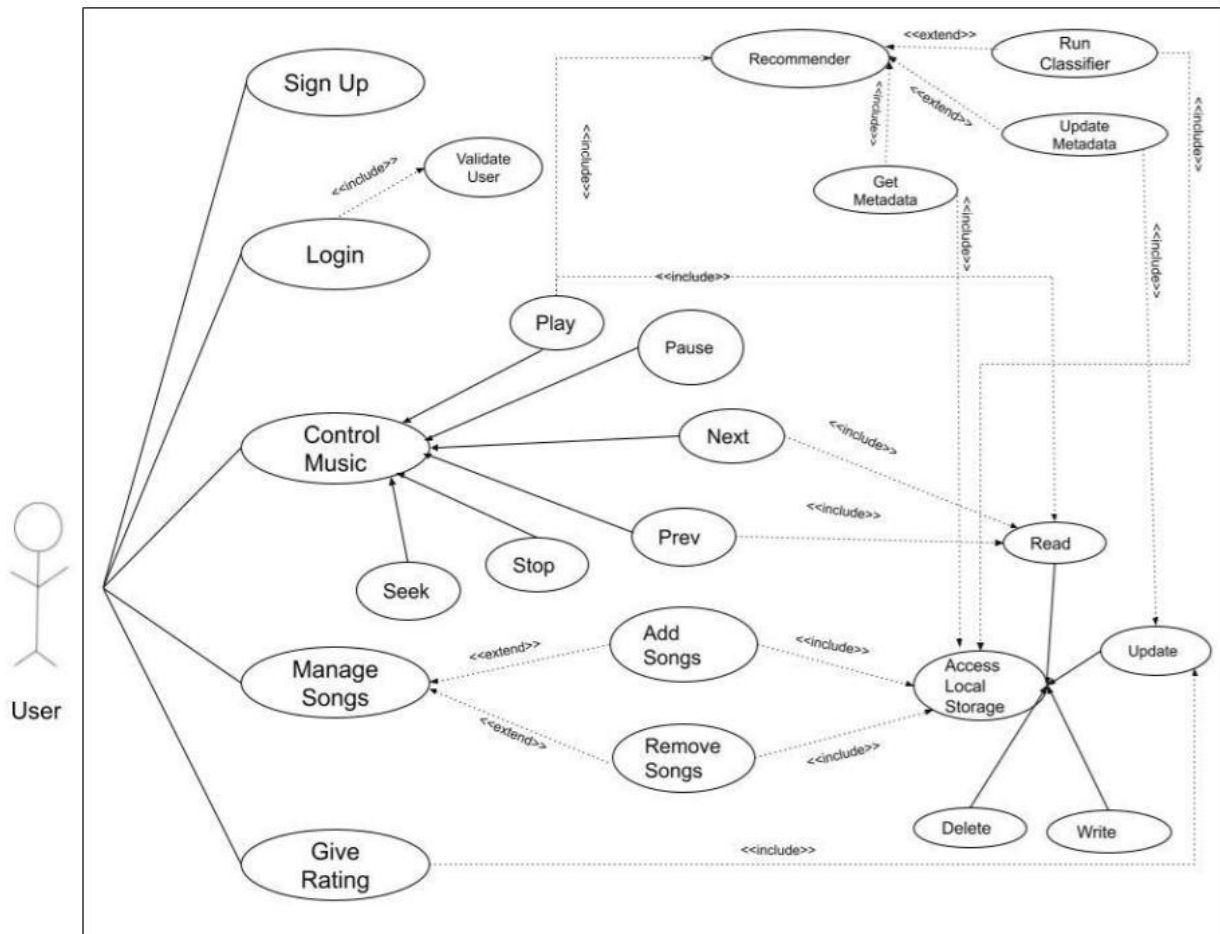
- For the front-end, the user interface is going to be designed using React.
- For the backend, sqlite is being used where we need to access songs and lyrics databases so the user can browse music.

3.3 Communications Interfaces

NAAS requires a web browser and an internet connection.

4. Analysis Models

4.1 USE CASE Diagram

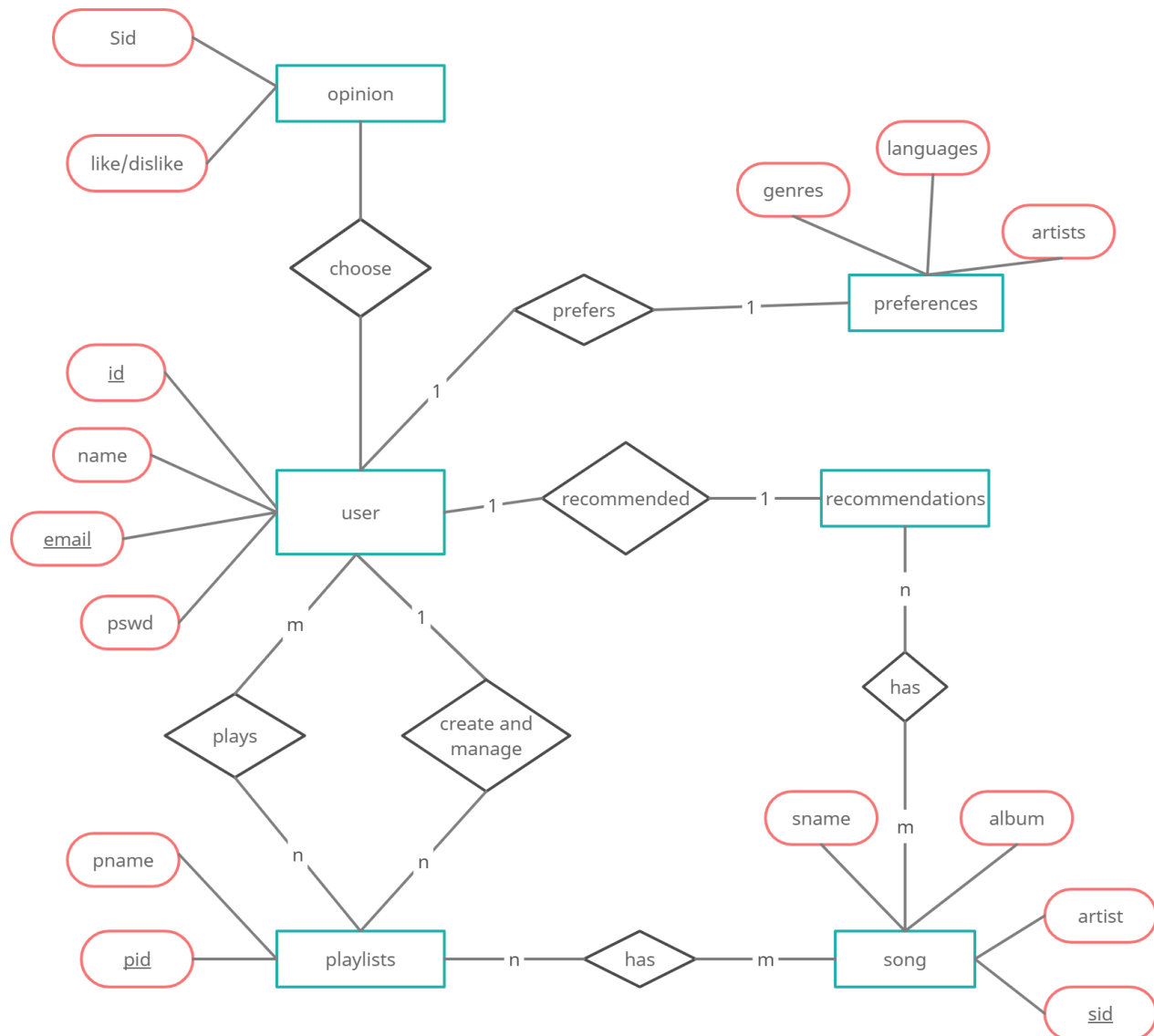


USE CASE DIAGRAM

Description -

- Name - Music Player App (NAAS)
- Summary - A music streaming application
- Actor - Anyone with good internet connectivity
- Pre conditions - No pre conditions
- Description - NAAS is a music streaming application in which a user can create an account, and while signing up they will be asked to select their favourite genres and singers based on which we will recommend the songs. A user can also manage songs by adding or deleting tracks from the playlist which will be saved in the local database of the user. Along with these features a user can also give ratings which will be used for improving the recommendations in the future.
- Exceptions - This application does not work offline
- Post-Conditions - The metadata is updated in the local storage

4.2 Entity-Relation Diagram of the database



5. System Features

5.1 Authentication(1)

5.1.1 Description and Priority

Users will be prompted to fill their details to sign up/login to avoid any kind of interventions while using the app. Having a personal account helps service providers gain a better understanding about the client and provide them with better service.

5.1.2 Stimulus/Response Sequences

When a user opens the app, they will be prompted to sign-up/login.

- For a user who already has an account can login through the user interface(UI) and fill their details. On proper authentication the user will be redirected to the home page.
- If a new user attempts to login a dialog box which says “No such user exists” pops up
 - Users will be redirected to the sign up page where users are expected to fill their details.
 - The app verifies that all the details provided are in the correct format to store it in the database.
 - The details will be stored in the database in the backend.
 - Then the user will be able to login to the app and will be navigated to the home page.

5.1.3 Functional Requirements

- Application should be launched in an environment with specified operating system requirements.
- Users are expected to enter valid details while signup such as a valid email id, phone number and make sure that the email id doesn't have an already existing account.
- In case the username and email id entered in the login page doesn't exist in the database, the user is considered a new user and is directed towards the signup page.

5.2 Search Songs(2)

5.2.1 Description and Priority

A search option is provided via UI to help users find any song from the database they would like to play.

5.2.2 Stimulus/Response Sequences

- Search icon seen in the toolbar at the bottom of the homepage directs the user towards the search UI(user interface), where the user can enter the name of the song, album or artist they want to listen to.
- The server searches the database and returns some songs/playlist which match the user query.

5.2.3 Functional Requirements

Users should enter the precise name of either the song, album or artist they intend to listen to. If the input string doesn't match any of the above categories, a song with a name similar to the entered name is displayed.


5.3 Managing Playlists(3)

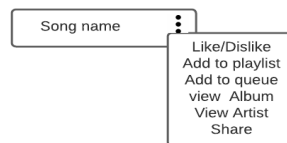
5.3.1 Description and Priority

Users are allowed to create multiple playlists where they can add their favourite tracks to the playlist to avoid searching. The playlist will be added to the user details in the database(backend) and gets updated as the user removes or adds tracks.

5.3.2 Stimulus/Response Sequences

User opens the app and goes to the library and creates a playlist. Currently chosen track is played utilizing the sound ports. Visual data about the track will be seen on the screen.

- Creating an empty playlist
 - User creates an empty playlist for the first time.
 - The system prompts the user for a name for the playlist. The playlist will be stored in the database(backend).
 - A playlist will be created.
- Add tracks to the playlist
 - User searches for a song.
 - Users can click on the “add to playlist icon”. 



- Users will be displayed with the list of playlist they have created in the past.
- Users can choose any playlist and the track will be added to the chosen playlist.
- The track will be added to the chosen playlist in the backend.

5.3.3 Functional Requirements


- Users must give a unique name to every playlist he creates.
- Each playlist has a unique set of songs i.e trying to add a song to a playlist in which it already exists is prohibited.

5.4 User ratings for the tracks(4)

5.4.1 Description and Priority

Users can give ratings to the tracks so that service providers can recommend songs to the users based on the ratings and the metadata(genre, artist, music etc.) of the rated songs.

5.4.2 Stimulus/Response Sequences

- Search for a song
- Click on the  icon.

- Rate the song as like/dislike  / 

5.4.3 Functional Requirements





TBD

5.5 Play/Pause/Next/Prev(5)

5.5.1 Description and Priority

Users can play/pause/stop songs according to their convenience. They can skip the song or listen to the previous song.

5.5.2 Stimulus/Response Sequences

- Search for a song.
- when the user plays the song, the user will have options to play/pause/stop.
 - Click on the play icon  to listen to the song.
 - Click on pause button  to pause the song.
 - Click on forward button  to skip the current track.
 - Click on backward button  to listen to the previous track.

5.5.3 Functional Requirements

- Users should select a song on which the above mentioned operations can be performed.

5.6 Recommendations (6)

5.6.1 Description and Priority

Based on user personal information and their previous activity in the application ,the songs to be displayed on the users homepage are decided. This helps users in discovering similar songs of interest.

5.6.2 Stimulus/Response Sequences

- On launch of the application the server decides which songs to display based on users preferences.

5.6.3 Functional Requirements

TBD

6. Other Nonfunctional Requirements

6.1 Performance Requirements

For a system to ought to be adequately quick to play music and react to any of the client activity in any capacity with no breaking or buffering, it requires a small amount of disk space for installation.

It can work well while other programs are running.

6.2 Safety Requirements

The software should warn the users when they go beyond threshold volume and do other things that are not suitable. The administrator cannot access personal information of the user, ensuring safety. The software would also have to warn the user to save their changed data before they exit.

6.3 Security Requirements

Connection between the NAAS server and the user must be encrypted.

6.3.1 Data Transfer

- System should automatically log out the user after being inactive for a certain period.
- System should not leave any cookies in the user's computer containing the password or any other confidential information.

6.3.2 Data Storage

User's web browser shall never display the user's password, instead it should be encoded using special characters, allowing it to be reset but never shown. Back-end servers shall only be accessible to authenticated administrators and databases should be encrypted.

6.4 Software Quality Attributes

- Memory Management - There should be no leakage of memory
- Error Handling
- Compatibility - The system should be compatible and peacefully exist together with other systems

6.5 Business Rules

- This software is an Open Source software

Other Business rules TBD

7. Other Requirements

7.1.1 Legal, Copyright and other notices.

Software must display all the applicable copyright.

Appendix A: Glossary

TBD - To be decided

Appendix B: Field Layouts

An Excel sheet containing field layouts and properties/attributes and report requirements.

Sample sheet with information required to Sign up to the app.

Field	Length	Data Type	Description	Is Mandatory
First Name	16	String		Y
Last Name	11	String		Y
Email-Id	20	AlphaNumeric		Y
Password	8	Alphanumeric		Y

Sample sheet with information required to store the tracks.

Field	Length	Data Type	Description	Is Mandatory
Song	20	String	Name of the song	Y
Song Id	8	Int	Unique identifier given to each song	Y
Artist	20	String	Name of the Artist	Y
Genre	20	String		N

Sample sheet with information required for playlist management

Field	Length	Data Type	Description	Is Mandatory
Email Id	20	AlphaNumeric		Y
Playlist	20	String	Name of the playlist	Y
Playlist Id	4	Int	Unique Identifier	Y

Sample sheet with information required to access users tracks

Field	Length	Data Type	Description	Is Mandatory
Playlist Id	4	Int	Unique Identifier	Y
Songs	100	Array	Array of Song Id's	Y

Appendix C: Requirement Traceability Matrix

Sl. No	Req. ID	Brief Description of Requirement	Architecture Reference	Design Reference	Code File Reference	Test Case ID	System Test Case ID
1	1	Authentication	-	-	-	-	-
2	2	Search Songs	-	-	-	-	-
3	3	Managing Playlist	-	-	-	-	-
4	4	Ratings	-	-	-	-	-
5	5	Play/Pause	-	-	-	-	-
6	6	Recommendations	-	-	-	-	-