**Challenge 1**

**Functional**

**The system must :**

1. Maintain a library of albums and songs
2. Allow users to browse albums and songs
3. Allow users to select individual songs
4. Prevent users from selecting entire albums
5. Maintain a queue of songs to play
6. Play music
7. Allow users to sort by artist
8. Identify individual users
9. Track number of plays per user

**Non-Functional**

**The system should be :**

1. Intuitive to use while floating in space
2. Available 24/7
3. Low-power
4. Updatable

**وظائف النظام**

**يقوم النظام ب:**

1. احتفظ بمكتبة من الألبومات والأغاني
2. السماح للمستخدمين بتصفح الألبومات والأغاني
3. السماح للمستخدمين بتحديد الأغاني الفردية
4. منع المستخدمين من اختيار البومات كاملة
5. احتفظ بقائمة انتظار من الأغاني لتشغيلها
6. تشغيل الموسيقى
7. السماح للمستخدمين بالفرز حسب الفنان
8. تحديد المستخدمين الفرديين
9. تتبع عدد مرات اللعب لكل مستخدم

**أساسي**

**يجب أن يكون النظام:**

1. سهل الاستخدام أثناء الطفو في الفضاء
2. متاح 24/7
3. طاقة منخفضة
4. قابل للتحديث

**Challenge 2**

**1 - use cases**

Title: play a song

Primary actor: User

Success Scenario: System identifies User.

User browses library of available albums.

User selects an album and browses list of songs on the selected albums.

User select a song.

system plays the selected song.

**2 - use case**

1. System identifies the users.
2. User browses the available albums and songs.
3. User selects a song.
4. system begins playing selected song.
5. The user continues browsing and selects a second song
6. system adds that second song to a play queue
7. system plays second song after the first song is over.

**user stories**

1. as a user, I want my song to be added to the front of the long play queue so that I don't have to wait hours to hear it.
2. as a user, I want to be identified without having to touch anything so I can use my hands to do other things.
3. As a user, I want to sort and browse songs by artist, so that I can listen to every song by ABBA.

**1- حالات الاستخدام :**

العنوان: تشغيل أغنية

الفاعل الأساسي: المستخدم

سيناريو النجاح: النظام يحدد المستخدم.

يستعرض المستخدم مكتبة الألبومات المتاحة.

يختار المستخدم ألبومًا ويتصفح قائمة الأغاني في الألبومات المحددة.

حدد المستخدم أغنية.

يقوم النظام بتشغيل الأغنية المختارة.

**2 - حالة الاستخدام :**

1. يحدد النظام المستخدمين.
2. يتصفح المستخدم الألبومات والأغاني المتوفرة.
3. يختار المستخدم أغنية.
4. يبدأ النظام في تشغيل الأغنية المختارة.
5. يستمر المستخدم في التصفح ويختار أغنية ثانية
6. يضيف النظام تلك الأغنية الثانية إلى قائمة انتظار التشغيل
7. يقوم النظام بتشغيل الأغنية الثانية بعد انتهاء الأغنية الأولى.

**قصص المستخدم :**

1. كمستخدم ، أريد إضافة أغنيتي إلى مقدمة قائمة انتظار التشغيل الطويلة حتى لا أضطر إلى الانتظار لساعات لسماعها.
2. كمستخدم ، أريد التعرف على هويتي دون الحاجة إلى لمس أي شيء حتى أتمكن من استخدام يدي للقيام بأشياء أخرى.
3. كمستخدم ، أرغب في فرز الأغاني وتصفحها حسب الفنان ، حتى أتمكن من الاستماع إلى كل أغنية بواسطة ABBA.

**Challenge 3**

**We identify things by looking at names in use cases and user stories:**

**Objects:**

* user
* library
* album
* song
* queue
* admin

**the responsibilities :**

* browse the library of album
* browses songs in the album
* select album
* select song
* play selected song
* add a song to the queue
* play a song from the queue
* cancel song in the queue



**نحدد الأشياء من خلال النظر في الأسماء في حالات الاستخدام وقصص المستخدمين:**

**أشياء:**

* المستعمل
* مكتبة
* الألبوم
* أغنية
* طابور
* مشرف

**المسؤوليات :**

* تصفح مكتبة الألبوم
* يتصفح الأغاني في الألبوم
* حدد الألبوم
* اختر أغنية
* تشغيل الأغنية المختارة
* إضافة أغنية إلى قائمة الانتظار
* تشغيل أغنية من قائمة الانتظار
* إلغاء الأغنية في قائمة الانتظار

**Challenge 4**

| **Admin** |
| --- |
| - id: String |
| + getld(): String  + createUser()  + manage Queue() |

| **Queue** |
| --- |
| - playlist: Songs[0..\*] |
| + addSong(Song, UserID)  + getNextSong(): Song  + remove Song(Song) |

| **User** |
| --- |
| - id: String |
| + getId(): String |

| **Song** |
| --- |
| - title: String  -artist: String |
| + getTitle(): String  + getArtist(): String  + play() |

| **Library** |
| --- |
| - titles: Strings[1..\*]  - albums: Albums[1..\*] |
| + getTitles(): Strings[1..\*]  + getAlbum(String): Album |

| **Album** |
| --- |
| - titles: Strings[1..\*]  - songs: Songs[1..\*] |
| + getTitles(): Strings[1..\*]  + getSong(String): Song |

**Challenge 5**

****