PRAC 1

Software Engineering

Alejandro Pérez Bueno

Apr 19, 2024

Table of Contents

Self-Responsibility Declaration

I understand that plagiarism, the use of AI or other generated content will imply that the delivered work will not be reviewed and it will be automatically assigned a grade of D. I certify that I have completed the PRAC1 individually and only with the help that the professors of this subject considered appropriate, according to the FAQs about plagiarism.

Question 1

Non-functional requirements:

Requirement	Description	Type	Stakeholder
The project must be developed at least in all of Spain's co-official languages, that is, in Catalan, Castilian, Basque, and Galician	The application interface and functionalities must be available in multiple languages to cater to the diverse linguistic preferences of podcasters across Spain	Cultural and policy	Ivan, podcaster
We always work using an agile methodology, with two-week sprints during which the whole team can see the project's progress and receive constant feedback	The development process should follow an agile approach with short iterations and frequent feedback loops to ensure adaptability and responsiveness to changing needs	Operational and environmental	Alex, software engineer
The most significant challenge for us will be managing very large files, so we'll likely need to integrate the platform with storage services like Amazon Web Services' S3 or equivalents	The system must be capable of managing and storing large files, possibly through integration with external storage services like Amazon Web Services' S3. This is necessary to handle the large audio files that will be uploaded by users	Maintainability and support	Alex, software engineer

Question 2

Functional requirements:

1. *As a **local radio director**, I want to enable or disable studios for rental based on the existing models to manage the availability of the studios.*
2. *As a **podcaster**, I want the platform to automatically upload and integrate my podcast recordings with popular platforms like Spotify, Google Podcasts, and Apple Podcasts to streamline the publishing process and reach a wider audience.*

3. *As a **sound technician**, I want to confirm my availability for a podcast recording session to manage my schedule effectively.*
4. *As a **software engineer**, I want to integrate the platform with storage services like Amazon Web Services' S3 to manage very large files.*

Question 3

a) Conflict in requirements

- **Stakeholders**: The software engineer and the podcaster.
- **Conflicting requirements**: The software engineer wants to use Amazon Web Services' S3 to manage very large files, while the podcaster wants the platform to automatically upload and integrate podcast recordings with popular platforms that upload podcasts. The conflict appears because meeting both requirements may lead to technical complexities such as:
  - Managing the upload and integration process across multiple platforms.
  - Ensuring data consistency.
  - Dealing with potential file size limitations on the podcast platforms.

b) Solution to the conflict

To solve the conflict, we could use AWS S3 as the primary storage for all our podcast files due to its scalability/reliability, and we could develop an automated process that converts and compresses the podcast recordings into a format and size acceptable by most if not all podcast platforms, while maintaining good audio quality.

Question 4

- Ideologue of the project (Mónica)
- Local radio director (Hatim)
- Sound technician (Carolina)
- Podcaster (Ivan)
- Software engineer (Alex)

Question 5

- **Use Case Identifier**: CU001
- **Main Actor**: Podcaster
- **Supporting Actors**: Studio Technician
- **Level**: User (user goals)
- **Scope**: Organization
- **Main Success Scenario**:

1. The Podcaster logs into the studio booking system.
2. The Podcaster selects the desired date and time slot for the studio rental.
3. The system checks the availability of the studio for the selected date and time slot.
4. The system confirms the availability of the studio.
5. The Podcaster confirms the booking.
6. The Studio Technician receives the booking request and confirms the booking.

- **Alternative Scenarios**:
  - 3a. If the studio is not available for the selected date and time slot, the system informs the Podcaster about the unavailability and prompts to select a different date or time slot.
  - 5a. If the Podcaster does not confirm the booking within a certain time frame, the system cancels the booking process automatically.
  - 6a. If the Studio Technician is unable to confirm the booking for any reason, the system informs the Podcaster about the situation and cancels the booking.

Question 6

a) User-level use cases from interviews

1. **Book a Podcast Recording Studio**: *We're looking to rent a studio periodically, like subscribing to a pass.*
2. **Modify a Booking**: *Of course, on days we host a guest, we'll need to modify our booking to request a larger studio.*
3. **Cancel a Booking**: *...we might have to cancel, and in those cases, I hope the cancellation fee is minimal.*

b) User-level use cases not mentioned in the interviews

1. **Manage Podcast Recordings**: This would involve actions like accessing past recordings, downloading audio files, and potentially editing metadata.
2. **Review and Rate Technicians**: Providing feedback on technicians would help build a reputation system and assist other podcasters in choosing suitable collaborators.

c) Task-level use cases

> **Note**
>
> For the use case *Book a Podcast Recording Studio*.

1. **Select Recording Date and Time**: The podcaster chooses the specific date and time slot they want to book the studio for.
2. **Choose Studio Size**: Based on the number of participants expected, the podcaster selects the appropriate studio size to accommodate everyone comfortably.