**Reflective Technical Report System feature List**

This report is a document on the development journey of my EventEase application. This document focuses on the practical implementation of its features, the selection and use of Azure services and other technologies as well as personal reflection on the learning process throughout this Project. (Microsoft, 2024a)

**Application feature list**

The EventEase application is a tool to help manage events, venues, bookings and provides a system for booking specialists.

EventEase main features are:

1. **Venue Management Functionality**

* The EventEase application supports full **CRUD** (Create, Read, Update, Delete) operations for managing venue data within the system. (Gemini, 2025)
* **Create**: Booking specialists can register new venues by inputting details such as the venue’s name, location, capacity, image URL, and its current availability status through the IsAvailable field.
* **Read (List & Details)**: Users can access a complete list of all venues stored in the system and view detailed information about each venue.
* **Update**: Existing venue records can be edited to make changes, including updates to their availability status and other attributes.
* **Delete**: Venues can be removed from the system. Except deletion is restricted if the venue is linked to any existing bookings, keeping data integrity.
* **Venue Availability Field**: Each venue record includes a Boolean field called IsAvailable which shows whether the venue is currently available for booking. (Microsoft, n.d.-a; Microsoft, n.d.-b)

1. **Event Management Functionality**

* The EventEase application includes comprehensive CRUD functionality. This functionality promotes the management and classification of event types by linking them to specific venues.
* **Create**: Booking specialists can create new event entries by providing details such as the event name, description, the associated EventType (e.g., Concert, Conference etc..) and the venue where the event will take place.
* **Read (List & Details)**: The system has a complete filterable list of all events, allowing users to sort events by EventType and view detailed information for each event. Including its assigned venue and classification.
* **Update**: Existing event information can be edited, including changes to the event's name, description, type, and related venue.
* **Delete**: Events can be deleted from the system. Though, deletion is restricted if the event is linked to existing bookings, safeguarding the booking record of the user.
* **Event Type Classification**: All events are classified using a EventType lookup, which helps consistency, filtering and reporting. (Microsoft, n.d.-a; Microsoft, n.d.-b)

1. **Booking Management Functionality**

* The EventEase application provides booking management capabilities, including full **CRUD** operations and advanced filtering features whilst concurrently preventing scheduling conflicts.
* **Create**: Booking specialists can create new bookings by linking them to an exact event and venue while stating the start, end dates and times. The system includes overlap detection to confirm that no two bookings conflict for the same venue and time.
* **Read (List & Details)**: All bookings can be viewed in a list that joins related venue and event data for better performance. Users can also access detailed information for each separate booking.
* **Update**: Existing bookings can be modified, including updates to the booking’s dates, status and other related details. Overlap detection safeguards are programmed to avoid scheduling conflicts.
* **Delete**: Bookings can be deleted from the system when no longer needed, allowing for well-organized data management.
* **Search Functionality**: Users can perform basic searches to locate bookings using either the BookingID or Event Name. (Microsoft, n.d.-a; Microsoft, n.d.-b)

1. **Advanced Filtering Options**:

* **Filter by Event Type**: This allows users to display bookings that are only related with a specific category of events, e.g., concerts or conferences.
* **Filter by Date Range**: This allows users to select a custom date and time range to view bookings that occur within that specified period.
* **Filter by Venue Availability**: Users can filter bookings based on availability status (IsAvailable) of the associated venue, helping identify which venues are open for future scheduling. (Microsoft, n.d.-a; Microsoft, n.d.-b)

1. **Error Handling and Validation**

* The EventEase application contains error handling and data validation. This guarantees reliability, data integrity and a user-friendly experience.
* **Client-Side Validation**: ASP.NET Core’s model binding and built-in data annotation attributes ([Required], [Range]) are used to validate user input directly in the browser. This provides feedback for missing or incorrectly formatted fields before the data is sent to the server.
* **Server-Side Validation**: Custom validation logic is applied within controller actions to handle more complex business rules. This includes validating date ranges and logic to prevent double-booking of venues which cannot be fully imposed on the client side alone. (Microsoft, n.d.-a; Microsoft, n.d.-b)
* **Deletion Restrictions**: The system imposes constraints that stops the deletion of venues or events that are currently associated with existing bookings. This safeguards and prevents accidental loss of crucial data.
* **User Notifications**: The application uses TempData to communicate success and error messages to users after actions e.g., creating a booking or attempting to delete a linked venue. These messages are designed to be clear and informative, for example: “Booking created successfully!” or “Cannot delete this venue as it has existing bookings.” (Gemini, 2025)

1. **Image Management and Application Architecture**

**Image Management with Azure Blob Storage**

* In the original phase of development, venue images were represented using static placeholder URLs. This was later enhanced by integrating Azure Blob Storage, allowing for scalable image handling.
* **Upload**: When creating or updating a venue, image files are uploaded to a chosen container within Azure Blob Storage.
* **Retrieval**: Stored images are retrieved via Azure Blob URLs, which improves both scalability and performance in content delivery.
* **Deletion**: When a venue is edited or deleted, outdated images are deleted from Blob Storage to prevent unnecessary storage use and maintain data consistency. (Gemini, 2025)

**Responsive User Interface**

* The application’s front-end is developed using the Bootstrap framework which guarantees a responsive and accessible user interface across multiple device types, desktops, tablets and smartphones. (Gemini, 2025)

**Data Persistence**

* All application data such as venues, events, bookings and event types are stored in a structured relational database that certifies data integrity and long-term persistence. (Gemini, 2025)

**Temporary Data Messaging**

* To improve user experience and feedback, the application pulls TempData to display related messages (e.g., confirmation or error messages) following user actions like creating, updating, or deleting records. For example: “Booking created successfully!” (Gemini, 2025).

**Deployed Application URL:**

**AZURE WEB APP URL HERE** –

[<https://eventeasewebappst10329226-amgrhyh6hsa9a0e5.southafricanorth-01.azurewebsites.net> ]

**Code Attribution and Referencing:**

**Microsoft. (n.d.-a). ASP.NET Core documentation. Microsoft Learn. Retrieved June 23, 2025, from** [**https://learn.microsoft.com/en-us/aspnet/core/**](https://learn.microsoft.com/en-us/aspnet/core/)

Microsoft. (2024a). What is Azure App Service? Azure Documentation. Retrieved June 23, 2025, from <https://learn.microsoft.com/en-us/azure/app-service/overview>

Microsoft. (2024c). Introduction to Azure Blob Storage. Azure Documentation. Retrieved June 23, 2025, from <https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction>

Smith, J. (2023, April 15). Database normalization explained: Why it matters for cloud applications. CloudGeek Blog. [Replace with an actual blog post or academic article if you used one for normalization concepts, otherwise use a generic source that covers it].

Microsoft. (2024b). Azure SQL Database overview. Azure Documentation. Retrieved June 23, 2025, from <https://learn.microsoft.com/en-us/azure/azure-sql/database/sql-database-paas-overview>

Microsoft. (n.d.-b). C# documentation. Microsoft Learn. Retrieved June 23, 2025, from <https://learn.microsoft.com/en-us/dotnet/csharp/>