Client History

Our client is The Old Globe Theatre, a community theater in San Diego - California. The theater was built in early 1935 for the presentation of abridged versions of Shakespeare's plays as part of the California Pacific International Exposition.

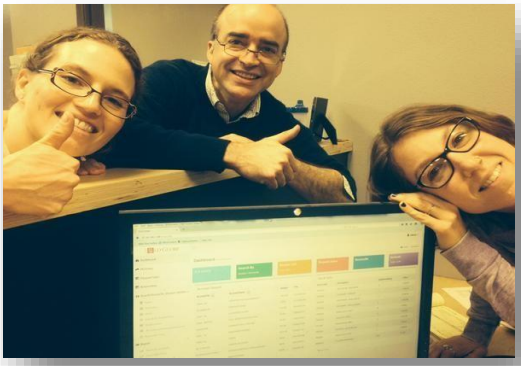
The theater has seen many remold, renovation, and new constructions since its inception. The development process is almost continuous. Taking care of accounts and stocks are the primary concern of the administration department.

Existing System Overview

To take care of accounts and stocks, they have developed an application using the Microsoft Access database. The system was believed to be developed around 1995. Different departments have used this application since then. They start getting delays in data entry and report generation due to Access data management limitations. This leads them to create a new system with a modern approach.

New Requirement Overview

The new system we developed after deeply analyzing their requirements and studying their various use-cases we developed a web application. We used PHP 5 for server-side scripting and MySQL 5.6 for data storage. The server was their existing Windows 2012 server.



How Our New Solution Has Performed?

We took precise care while analyzing the requirements and developing each module. The only intention was to make the system work faster, better reporting, simultaneous user transactions, faster search-data fetching, and desktop-like user experience on web applications using keyboard shortcuts. Our new solution seems to have worked nicely and has improved their workflow. The client has posted an image showing our success!

Existing Business Application & Problems

The administration team has different locations around the campus. They have different departments like stock keeping is done from the warehouse and accounting from the main administration office. These departments were connected via internal network and internet if someone managed the data remotely.

They have developed their own accounting system on top of Microsoft Access and have been using it since 1995. The system was a good fit for earlier days, where team size is relatively small, and the system is lighter on the data storage part. The system was divided into two functional layers one for the front-end user interaction part, and the other was the database part.

They start facing the problem as their database starts growing, the team starts expanding and the requirement of simultaneous user transactions. Access is not an RDBMS fundamentally, so it lacks modern age database features like indexing, key relationships, views creation, full-text search, and modern performance tuning.

As their data grew significantly, the module load time was getting higher. Some modules were taking minutes to load for user input. They saw a huge delay when the application database on another department server and the user was connected via a local network. Finding records was taking more time as access needed to search through a huge bulk of records. Reporting was their most-used module, and it was taking more time to pull data and run reporting business logic on it to prepare the report.

So when they contact us for new system development, they have a condition that all their existing data should be migrated to a new solution. They gave us a complete walkthrough of their existing system. Then give us an overview of all the problems they were facing.

How we Approach the Problem with the Solution Which we Develop

We started analyzing their requirements. We used their current application on our server to understand it better. We asked them about the use-cases and business process flow. We listened and logged every piece of information they provided and the request for new/extra features they need.

We targeted their use-case and problems. And based on our analysis and discussion with the client, we found that they need an application that covers the below points.

- **Central Database**
- **Open Source-technology**
- **Simultaneous Multiple User Transaction**
- **Easy and Intuitive Front-End User Experience**
- **Easy Deployment**
- **Faster Data-Processing**
- **Customized, Easy and Printable Reports**

Now it was our task to choose the right framework and software stack that satisfies the mentioned requirements. We choose Web Application in this case. We may have gone for Windows Desktop applications as they were already using Windows Server 2012. Considering reporting, heavy usage of charts, portability, device independence, and OS independent, we decided to go for a web application path.

Based on our research, we thought of dividing the entire application into 3 distinct ways.

1. **Business Logic and Data Storage Layer**
2. **Web Front-End Layer**
3. **Service Layer for Asynchronous Requests**

Development Process:
The development process is divided into different sections ensuring to provide our clients excellent solution as per their specified requirements:

I. **Analysis and Requirement Finalization**
II. **Project Management**
III. **Communication**
IV. **Design**
V. **Technical Details**

Analysis and Requirement Finalization:

We took enough time to analyze their requirements. There were some areas that were use-case and organization-specific. We asked for use-cases to know how they operate modules and how they would like those modules to behave on a new application.

Project Management:

After requirement gathering and features finalization, we started working on the system. We choose an agile development methodology. We divided the system into a set of modules and started working on the core (most independent) module to a specific (most dependent) module. We used Trello for managing our project. We invited our clients on board so they can see the work progress in real-time and provide feedback in a real-time manner.

Project management is one of the fundamental core elements that make development efficient and puts our client and us on the same page, an open system where the client can see team working projects, and each member working on module-specific tasks. As we develop and test modules. We ask our client to finally test the module against the real-world case. After testing, he can directly notify the team-member about the update.

Communication:

An analysis is a part where communication is heavily used. Once the analysis is completed, we used to talk to our client when a new module developed, and we needed to showcase the update on a real-time screen-share basis. Or times when the client wants to make some part of

the application clear to us. The project management is efficient enough that the client needed to talk to us once or twice during the development life cycle for roughly 8 months.

Design:

Design is the first and foremost important as users spend more time working with the front-end of the system. We used a Bootstrap-based responsive design that looks uniform across devices. The design theme was HTML5 and CSS3 based. Their concern was the web application should behave the same way as a desktop app. To achieve this, we heavily used jQuery for front-end engineering.

Technical Specification

As we stated, we choose the web application path for the central database, easy deployment, and device and OS independence it offers. We choose the open-source technology stack that would not require our client to purchase additional licenses. We selected PHP for server scripting and MySQL for database management.

Their primary intention was the new web application should work fast. So we designed the proper relational schema on MySQL and indexed the required entities. We designed the code that mixes asynchronous and synchronous requests to the server to balance the load and faster data processing.

We utilize Git for version management. As we are working in a team, it is essential to use version management for better collaboration. The release branch manages the full version of the release-ready code. We maintain the same version on the production server and GIT production branch.