Starbase: A Worklog Application

Abstract

Starbase is a worklog application migrated from Microsoft Access to a Java-based platform due to performance limitations. The transition enhances speed, scalability, and reliability while maintaining core functionalities. The new Java application allows customization of displays/features and optimizes worklog efficiency ensuring a smoother and more responsive management experience.

Development Process

1. Gather functionality from Microsoft Access application and requirements from stakeholders
2. Determine what platform to use (Python's Streamlit vs Java)
3. Create a prototype of an individual page for approval
4. Develop UI design and features for each screen
5. Test each screen's functionality
6. Push to production and maintain a backlog of defects and enhancements

Programming Languages and Software's

Java
JavaScript
JavaServer Pages
CSS
SQL
JBoss EAP
Microsoft SQL Server
Visual Studio Code
GitHub

Application Requirements

- Default user is set to the associate opening the application
- Only associates granted access to the application can view
- Ability to change user
- Ability to view current user active work items
- Ability to view current user support tasks
- Ability to view all work items
- Ability to view all support tasks
- Ability to add a work item
- Ability to add a support task
- Ability to edit a work item
- Ability to edit a support task
- Ability to add a note to a work item
- Ability to plan/track work item hours by week
- Ability to filter on all pages that display work items
- Ability to sort on all pages that display work items
- Ability to export all work items
- Ability to export all support tasks
- Conditional formatting in columns of display pages
- Input validation on data entry

Results

This project led to a full-scale application in production to be used by teams within the IT Business Intelligence and Data Management department at The Cincinnati Insurance Company. This application is being maintained through a backlog that holds defects and enhancements to be addressed in future versions. All features from the Microsoft Access version were transitioned to the Java based application in addition to new features such as filtering and exporting of data. The Java based Starbase application addressed two main concerns of Microsoft Access Starbase being slow and challenging for the user. Correcting these concerns allowed for a better user experience and higher participation.

Design Decisions

Throughout the process of developing the Starbase application, there were many decisions to make to create the most efficient and usable application that met stakeholder needs. Some of these decisions included choosing the correct platform for the application, how to set and maintain the user, and creating the most efficient filtering pattern.

Python's Streamlit vs Java

The first decision was to determine which platform to use Streamlit or Java code. Both structures had been used internally for previous dashboards; however, it was necessary to determine which would be the best for a full-scale application. Streamlit allowed for quick development but did not have customizability in the UI and interactivity. Streamlit also began to run slower when the amount of data increased. Java allowed all requirements to be met along with efficient data extraction and customizability in UI and interactivity but took longer to develop. The time it would take to work around the downfalls of Streamlit made it an easy decision to use Java.

Initializing the User

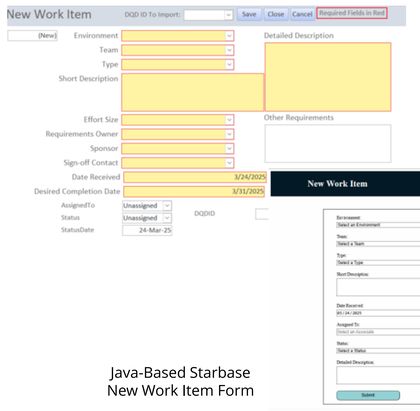
One functional requirement is the application opening to the user accessing it. The known pattern was getting the user's associate ID from stored cookies and then calling a SQL Stored Procedure to get the associate's full name. This added a call to SQL Server that was not needed as the application was already making an API call to ensure the user had permission to access that application. A minor change was made to the API call to include the associate's full name along with the permissions they have.

Filtering Pattern

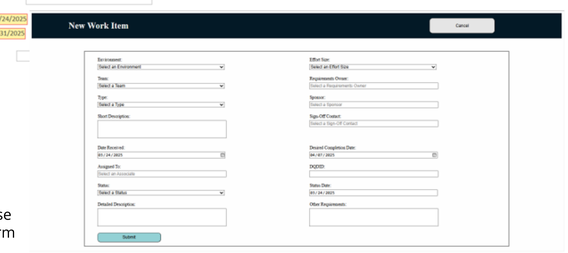
A new requirement after the initial push to production was having filtering and sorting on all pages that display work items. Initially for each page a SQL Server Stored procedure was called with different parameters based on user selection. This created hard to follow stored procedures and lack of scalability. The initial pattern only allowed for one column to be filtered on with a single value and one column to be sorted on at a time. A new pattern needed to be created which led to multiselect filtering along with multiple columns. This pattern created the SQL query in the Java code along with filtering/sorting values stored in a session 2D array. The SQL query would then be sent to SQL Server with the return being displayed. This pattern caused some initial pains with how to store the variables between front-end and back-end code. It was determined to send the values being changed to the back-end which held the session variable. The pattern was slow at first due to the creation of the filtering options which was done be looping through all the values in the data. The speed was enhanced by calling SQL Server Stored Procedures that returned the values to be filtered from.

Microsoft Access Starbase Home Page

Java-Based Starbase Home Page

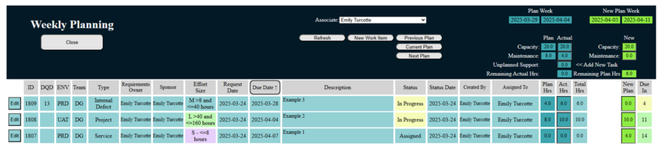


Microsoft Access Starbase New Work Item Form



Java-Based Starbase New Work Item Form

Microsoft Access Starbase Weekly Planning Page



Java-Based Starbase Weekly Planning Page

Emily Turcotte
Bachelor of Science, Computer Science
Minor in Business Analytics
Minor in Mathematics

Sponsor: The Cincinnati Insurance Company, IT Business Intelligence and Data Management