Independent Study Final Report

John Luukkonen

5/20/2021

This independent study course was focused on developing a full-stack application using the Vaadin framework, Java, and MS Sql database on the back-end. Study began by completing a tutorial to learn the Vaadin framework, then an original application was designed and built for the remainder of the term. The project is completed with the following deliverables. Dated Development Journal, Final Report(this document), Final presentation and Slide Deck submitted, Course Retrospective, Vaadin Tutorial CRM Application Source Code, and the Student-defined Application Source Code.

The Vaadin tutorial was provided as a marketing Ebook for free from Vaadin. This CRM Application allowed a user to view the list of current contacts, edit an existing contact, delete an existing contact, and enter a new contact which can be saved to the database. The dashboard view also contained a graphic displaying the percentage of contacts from each of the associated vendors. This was displayed as a pie chart, the graphic visualization is one of the tools included in the Vaadin framework.
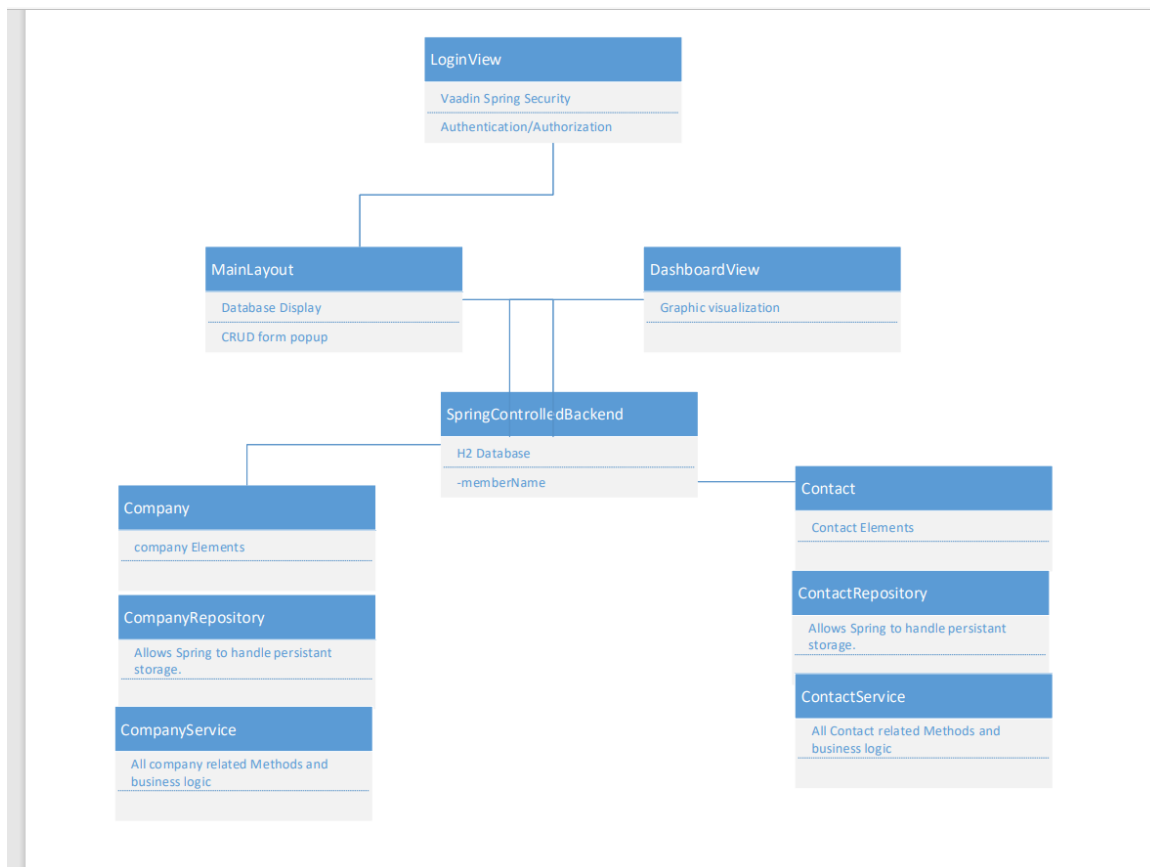
The application begins with the front end "views" these are written in Java referencing the many tools and libraries provided by Vaadin. The views include LoginView, MainLayout, and DashboardView. Much of the backend processes are controlled by Spring which exists as a bit of a "black box." For instance the H2 database is implemented automatically by use of Spring and the Java Persistence API. The Company and Contact classes have accompanying CompanyRepository and ContactRepository. These objects are marked for persistent storage through annotations in the code. The CompanyService and ContacdtService provide all business logic and to build the databases for each class in the tutorial.
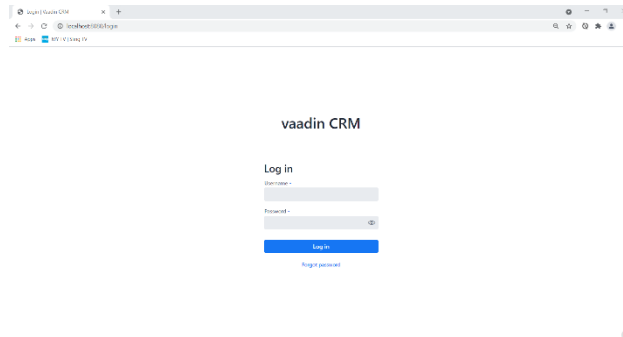
Landing page is for Authentication and Authorization.
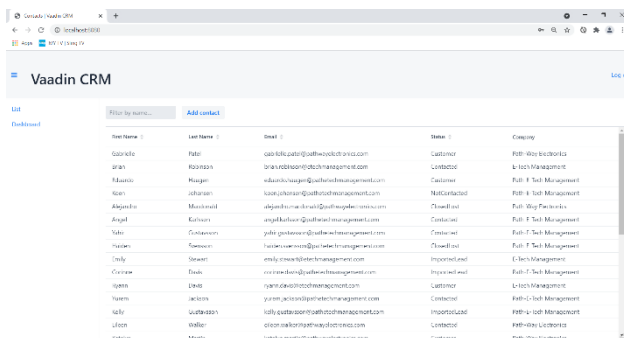
The MainView displays the content of the database. In this application that is a list of contacts. On the left side there are links to the "List" which is the current view, and Dashboard, which is a graphic display. Single click on a record will load that record in to the edit screen fields for update or delete. Clicking the Add Contact button will open the edit screen empty, ready for a New Contact.
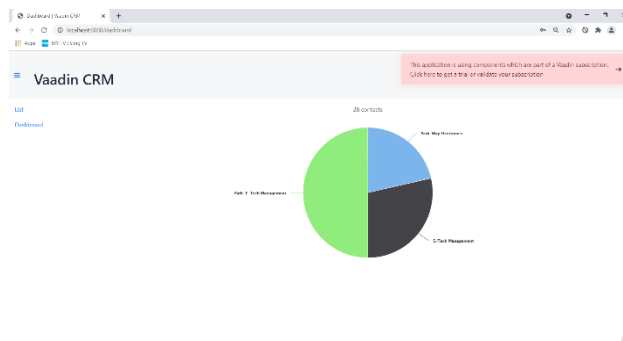
The Dashboard displays the percentage of contacts associated with each of the vendors listed.

I was inspired by the Dungeons and Dragons Beyond website to create a tool to support the Star Wars Tabletop Roleplaying Game. This discontinued D6 based game has a strong following online and this tool seems like it may provide some value to those fans. The application will perform the CRUD operations and store characters in a relational database to be recalled, updated, or deleted. I began with the Vaadin tutorial app as a start. I removed most of the elements and built my application upon the existing structure. I used Vaadin primarily to build the front end. I decided to implement an MS SQL database and access it using JDBC instead of Spring and JPA from the tutorial as an additional challenge and opportunity to gain experience with JDBC and database administration.
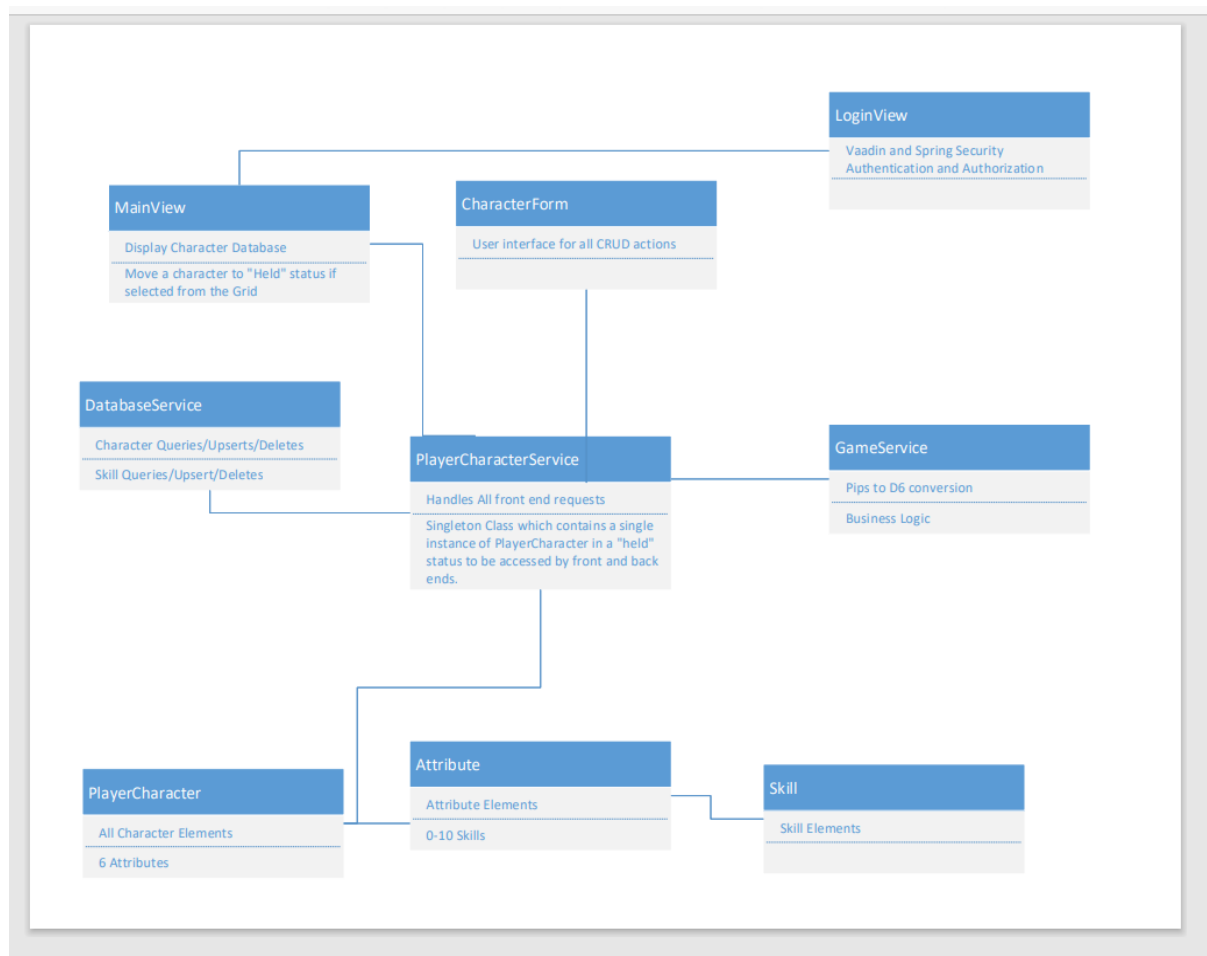
A user will log in to the application. A list of characters in the database will be displayed and accessible by the user. The user may also create a new character and save it to the database. The create, recall, update, and delete functions will happen in the CharacterView screen, a form with fields for all the data on a character sheet.

User will log in on the landing page, this handles Authentication and Authorization.

The MainView displays the characters in the database and offers users the opportunity to select an existing character to view/edit/delete in the CharacterForm, or by clicking New Character, navigate to a blank CharacterForm to create a new Character.

The CharacterForm is a form which has fields for all elements of a Star Wars RPG character sheet. Users can enter any or all data, including selecting

Attribute Skills from the drop-down menus and adjust Attribute levels . From this screen characters can be created, viewed, updated, or deleted.