```java
package fi.paatti.research.paattiapplication.components;


import com.vaadin.data.Container.Filter;
import com.vaadin.data.util.filter.Compare;
import com.vaadin.data.util.filter.Like;
import com.vaadin.data.util.sqlcontainer.RowId;
import com.vaadin.data.util.sqlcontainer.SQLContainer;
import com.vaadin.event.ItemClickEvent;
import com.vaadin.ui.Button.ClickEvent;
import com.vaadin.ui.Button.ClickListener;
import com.vaadin.ui.*;
import fi.paatti.paattidatabaseutils.dbservice.querydelegates.PaattiQueryDelegate;
import fi.paatti.paattidatabaseutils.names.PaattiColumnNames;
import fi.paatti.paattidatabaseutils.names.PaattiTableNames;
import fi.paatti.research.paattiapplication.ElementNames;
import fi.paatti.research.paattiapplication.views.ApplicationView;


/**
 * The search component is used to search adequate database items in various
 * views.
 *
 * @author Tapio Keränen, t.tapio.keranen@student.jyu.fi
 * @author Toni Salminen, toni.a.j.salminen@student.jyu.fi
 */
public class SearchComponent extends VerticalLayout {

  private ApplicationView applicationView;
  private String queryTableName;
  private String queryTableIdColumn;
  private String[] visibleColumns;
  private TextField searchField;
  private CheckBox toggleShowDeleted;
  private Table searchQueryContents;
  private Button createButton;
  private Button searchButton;
  private Button selectButton;
  private Filter searchFilter;
  private Filter activeRowFilter;
  private SQLContainer queryResults;

  public static final String[] USER_COLUMNS = new String[] { PaattiColumnNames.USER_userID, PaattiColumnNames.USER_name};
  public static final String[] GROUP_COLUMNS = new String[] {
    PaattiColumnNames.USERGROUP_usergroupID,
    PaattiColumnNames.USERGROUP_name};
```

```
  public static final String[] SCHEDULE_COLUMNS = new String[] {
    PaattiColumnNames.SCHEDULE_scheduleID,
    PaattiColumnNames.SCHEDULE_description};
  public static final String[] RESEARCH_COLUMNS = new String[] {
    PaattiColumnNames.RESEARCH_researchID,
    PaattiColumnNames.RESEARCH_name};
  public static final String[] EVENT_COLUMNS = new String[] {
    PaattiColumnNames.EVENT_eventID,
    PaattiColumnNames.EVENT_title};

  /**
   * SearchComponent constructor.
   *
   * @param view the view that owns this component instance.
   * @param tableName the name of the query target table.
   * @param tableIdColumn the name of the table's id column.
   */
  public SearchComponent(ApplicationView view, String tableName, String tableIdColumn) {
    applicationView = view;
    queryTableName = tableName;
    queryTableIdColumn = tableIdColumn;
    visibleColumns = getVisibleColumns(tableName);

    searchField = new TextField(ElementNames.SEARCH_COMPONENT_SEARCH_FIELD_LABEL);
    searchField.setWidth("100%");
    searchField.setEnabled(true);

    toggleShowDeleted = new CheckBox(ElementNames.SEARCH_COMPONENT_TOGGLE_SHOW_DELETED, false);

    searchQueryContents = new Table();
    searchQueryContents.setSelectable(true);
    searchQueryContents.setSizeFull();
    searchQueryContents.setEnabled(false);
    searchQueryContents.setNullSelectionAllowed(false);
    searchQueryContents.addListener(new ItemClickEvent.ItemClickListener() {
      public void itemClick(ItemClickEvent event) {
        if (event.isDoubleClick()) {
          setViewContents();
        }
      }
    });
    createButton = new Button(ElementNames.SEARCH_COMPONENT_NEW_BUTTON_LABEL);
    createButton.addListener(new ClickListener() {

      public void buttonClick(ClickEvent event) {
        applicationView.setContents(null);
```

```
        searchQueryContents.setValue(null);
        selectButton.setEnabled(false);
      }
    });
    createButton.setWidth("100%");
    createButton.setHeight("30px");

    searchButton = new Button(ElementNames.SEARCH_COMPONENT_SEARCH_BUTTON_LABEL);
    searchButton.addListener(new ClickListener() {

      public void buttonClick(ClickEvent event) {
        searchQueryContents.setEnabled(true);
        selectButton.setEnabled(true);
        refresh(null);
      }
    });
    searchButton.setWidth("100%");
    searchButton.setHeight("30px");

    selectButton = new Button(ElementNames.SEARCH_COMPONENT_SELECT_BUTTON_LABEL);
    selectButton.addListener(new ClickListener() {

      public void buttonClick(ClickEvent event) {
        setViewContents();
      }
    });
    selectButton.setEnabled(false);
    selectButton.setWidth("100%");
    selectButton.setHeight("30px");

    addComponent(createButton);
    addComponent(searchField);
    addComponent(toggleShowDeleted);
    addComponent(searchButton);
    addComponent(searchQueryContents);
    addComponent(selectButton);

    setExpandRatio(createButton, 0);
    setExpandRatio(searchField, 0);
    setExpandRatio(toggleShowDeleted, 0);
    setExpandRatio(searchButton, 0);
    setExpandRatio(searchQueryContents, 1);
    setExpandRatio(selectButton, 0);

    setSizeFull();
```

```
    activeRowFilter = new Compare.Equal(tableName + "_rowStatus", PaattiQueryDelegate.ROWSTATUS_ACTIVE);

    fillSearchTable();
  }

  /**
   * Returns the visible columns matching the query table.
   */
  private String[] getVisibleColumns(String tableName) {
    if (tableName.equals(PaattiTableNames.USER)) {
      return USER_COLUMNS;
    } else if (tableName.equals(PaattiTableNames.USERGROUP)) {
      return GROUP_COLUMNS;
    } else if (tableName.equals(PaattiTableNames.SCHEDULE)) {
      return SCHEDULE_COLUMNS;
    } else if (tableName.equals(PaattiTableNames.RESEARCH)) {
      return RESEARCH_COLUMNS;
    } else if (tableName.equals(PaattiTableNames.EVENT)) {
      return EVENT_COLUMNS;
    }
    return null;
  }

  /**
   * Fills the search table.
   */
  private void fillSearchTable() {
    queryResults = applicationView.getDBConnection().getSQLContainerFromDBTable(queryTableName, queryTableIdColumn, true);

    searchQueryContents.setContainerDataSource(queryResults);
    searchQueryContents.setVisibleColumns(visibleColumns);
    searchQueryContents.setColumnExpandRatio(visibleColumns[0], 0);
    searchQueryContents.setColumnExpandRatio(visibleColumns[1], 1);
    searchQueryContents.setEnabled(true);

    selectButton.setEnabled(true);
  }

  /**
   * Sets the filter for the search.
   */
  private void setSearchFilter() {
    queryResults.removeContainerFilter(searchFilter);
    queryResults.removeContainerFilter(activeRowFilter);

    searchFilter = new Like(visibleColumns[1], "%" + searchField.getValue().toString().trim() + "%", false);
```

```
    queryResults.addContainerFilter(searchFilter);

    if (!toggleShowDeleted.booleanValue()) {
      queryResults.addContainerFilter(activeRowFilter);
    }
  }

  /**
   * Refreshes the component's searchQueryContents table by fetching new
   * contents into the table from the database.
   *
   * @param id The row's id that should be set as the selected value. If null,
   * no value is selected.
   */
  public void refresh(Object id) {
    setSearchFilter();
    setSelected(id);
    setViewContents();
  }

  /**
   * Sets the row with the given id as the selected row in searchQueryContents
   * table.
   *
   * @param id The row's id.
   */
  private void setSelected(Object id) {
    if (id == null) {
      return;
    }

    RowId itemId = (RowId) searchQueryContents.firstItemId();

    while (itemId != null) {
      if (itemId.toString().equals(id.toString())) {
        searchQueryContents.setValue(itemId);
        searchQueryContents.setCurrentPageFirstItemId(itemId);
        break;
      }
      itemId = (RowId) searchQueryContents.nextItemId(itemId);
    }
  }

  /**
   * Sets the content of the parent view.
   */
```

```
  private void setViewContents() {
    if (searchQueryContents.getValue() != null) {
      applicationView.setContents(searchQueryContents.getItem(searchQueryContents.getValue()));
    }
  }
}
```