```java
package fi.paatti.paattidatabaseutils.dbservice.querydelegates;


import com.vaadin.data.util.sqlcontainer.RowItem;
import com.vaadin.data.util.sqlcontainer.query.generator.StatementHelper;
import fi.paatti.paattidatabaseutils.names.PaattiColumnNames;
import fi.paatti.paattidatabaseutils.names.PaattiTableNames;
import java.io.Serializable;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;


/**
 * Delegate for getting all voluntary events.
 * <p/>
 * @author Lauri Satokangas, lauri.n.satokangas@student.jyu.fi
 * @since 0.0.3
 */
public class VoluntaryEventsDelegate extends PaattiQueryDelegate implements Serializable {

  private static final long serialVersionUID = 1L;
  private final Object userID;

  /**
   * Constructor.
   * <p/>
   * @param userID The id of the user whos events will be searched.
   */
  public VoluntaryEventsDelegate(Object userID) {
    super(PaattiTableNames.EVENT, PaattiColumnNames.EVENT_eventID,
        PaattiColumnNames.getColumnNamesInArray(PaattiTableNames.EVENT));
    this.userID = userID;
  }

  /**
   * {@inheritDoc }
   * <p/>
   * Not in use.
   */
  @Override
  public int storeRow(Connection conn, RowItem row) throws SQLException {
    throw new UnsupportedOperationException("Not supported yet.");
  }

  /**
```

```
   * {@inheritDoc }
   * <p/>
   * Not in use.
   */
  @Override
  protected void setRowValues(PreparedStatement statement, RowItem row) throws SQLException {
    throw new UnsupportedOperationException("Not supported yet.");
  }

  /**
   * Returns a new StatementHelper containing the SELECT query statement. Gets
   * all the events that have -1 as  their eventTimeID (eg. there is no
   * assigned event time), that are assigned to the user and have active row
   * status.
   *
   * @param offset the starting row.
   * @param limit the number of rows to get.
   * @return The StatementHelper containing the SELECT query.
   * @throws UnsupportedOperationException
   */
  public StatementHelper getQueryStatement(int offset, int limit) throws UnsupportedOperationException {
    StatementHelper queryHelper = new StatementHelper();
    StringBuilder querySb = new StringBuilder();

    querySb.append("SELECT * FROM " + PaattiTableNames.EVENT);
    querySb.append(" WHERE " + PaattiColumnNames.EVENT_EVENTTIME_eventTimeID + " = '-1'"); // -1 means that there is no eventtime ⬀
assigned to the event, therefore the event is voluntary.
    querySb.append(" AND ").append(PaattiColumnNames.EVENT_rowStatus).append(" = '"⬀
).append(PaattiQueryDelegate.ROWSTATUS_ACTIVE).append(
        "'");
    querySb.append(" AND ").append(PaattiColumnNames.EVENT_SCHEDULE_scheduleID).append(" IN ");
    querySb.append("(SELECT ").append(PaattiColumnNames.SCHEDULE_scheduleID).append(" FROM ").append(
        PaattiTableNames.SCHEDULE);
    querySb.append(" LEFT OUTER JOIN (SELECT ").append(PaattiColumnNames.BELONGS_USERGROUP_usergroupID).append(" AS 'userGroupID' ⬀
FROM ").append(
        PaattiTableNames.BELONGS);
    querySb.append(" WHERE ").append(PaattiColumnNames.BELONGS_USER_userID).append(" = ").append(userID).append(
        ")  BELONGS_UG");
    querySb.append(" ON ").append(PaattiColumnNames.SCHEDULE_USERGROUP_usergroupID).append(" = userGroupID WHERE "⬀
).append(PaattiColumnNames.SCHEDULE_USERGROUP_usergroupID).append(
        " = userGroupID )");
    queryHelper.setQueryString(querySb.toString());
    return queryHelper;
  }
}
```