```
/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package view.studentView;

import java.io.IOException;
import static java.lang.Math.E;
import java.net.URL;
import java.time.LocalDate;
import java.time.LocalDateTime;
import java.time.ZoneId;
import java.time.ZoneOffset;
import java.util.Date;
import java.util.List;
import java.util.ResourceBundle;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.fxml.Initializable;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.RadioButton;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.scene.control.TextField;
import javafx.scene.control.cell.PropertyValueFactory;
import javafx.scene.text.Text;
import javafx.stage.Stage;
import model.ActiveUser;
import model.BorrowedEquipments;
import model.Equipments;
import presenter.ActiveUserPresenter;
import presenter.BorrowedEquipmentPresenter;
import presenter.EquipmentPresenter;
import presenter.StudentPresenter;

/**
 * FXML Controller class
 *
 * @author Adarsha
 */
public class StudentHistoryController implements Initializable {

    @FXML
    private Button notify_fx;
    @FXML
    private Button homeadmin_fx;
    @FXML
    private Button equipmentSearch_fx;
```

```
    @FXML
    private Text home_fx;
    @FXML
    private TableView<BorrowedEquipments> equipdettable_fx;
    @FXML
    private TableColumn<BorrowedEquipments, String> id_column;
    @FXML
    private TableColumn<BorrowedEquipments, String> name_column;
    @FXML
    private TableColumn<BorrowedEquipments, String> brand_column;
    @FXML
    private TableColumn<BorrowedEquipments, String> type_column;
    @FXML
    private TextField equipsearch_fx;
    @FXML
    private Button searchequipdet_fx;
    @FXML
    private Button fullEquipmentListBtn;
    @FXML
    private TableColumn<BorrowedEquipments, String> status_column;
    @FXML
    private Button returnEquipmentBtn;

    private BorrowedEquipmentPresenter borrowedEquipmentPresenter;
    @FXML
    private TableColumn<BorrowedEquipments, Date> borrowDate_column;
    @FXML
    private TableColumn<BorrowedEquipments, Date> returnDate_column;

    private ActiveUserPresenter activeUserPresenter;

    private EquipmentPresenter equipmentPresenter;
    @FXML
    private TableColumn<BorrowedEquipments, Integer> borrowId_column;

    private StudentPresenter studentPresenter;
    @FXML
    private Button searchequip_fx;
    @FXML
    private Button booking_fx;




    /**
     * Initializes the controller class.
     */
    @Override
    public void initialize(URL url, ResourceBundle rb) {
        borrowedEquipmentPresenter = new BorrowedEquipmentPresenter();
        activeUserPresenter = new ActiveUserPresenter();
        equipmentPresenter = new EquipmentPresenter();
        studentPresenter = new StudentPresenter();

        List <BorrowedEquipments> borrowedEquipmentList =
borrowedEquipmentPresenter.getAllBorrowedEquipments();
```

```
        getAllBorrowedEquipmentList((List<BorrowedEquipments>)
borrowedEquipmentList);

    }

    @FXML
    private void onHomeButtonClicked(ActionEvent event) throws
IOException {
        Parent root =
FXMLLoader.load(getClass().getResource("StudentDashboard.fxml"));
        Scene scene = new Scene(root);
        Stage window = (Stage) ((Node)
event.getSource()).getScene().getWindow();
        window.setScene(scene);
        window.show();
    }

    @FXML
    private void onSearchButtonClicked(ActionEvent event) {

            String keyword = equipsearch_fx.getText();
            System.out.println("Keyword is :"+ keyword);
            List<BorrowedEquipments> searchedEquipments =
borrowedEquipmentPresenter.findEquipmentsByName(keyword);
            getAllBorrowedEquipmentList(searchedEquipments);

    }

    @FXML
    private void onFullEquipmentListBtnClicked(ActionEvent event) {
        List <BorrowedEquipments> borrowedEquipmentList =
borrowedEquipmentPresenter.getAllBorrowedEquipments();
        getAllBorrowedEquipmentList((List<BorrowedEquipments>)
borrowedEquipmentList);

    }

    @FXML
    private void onReturnEquipmentBtnClicked(ActionEvent event) {
        BorrowedEquipments borrowEquipments =
equipdettable_fx.getSelectionModel().getSelectedItem();
        if(borrowEquipments != null){
            int equipmentId = borrowEquipments.getEquipmentID();
            Date borrowDate =
equipdettable_fx.getSelectionModel().getSelectedItem().getBorrowDate();
            Date returnDate =
equipdettable_fx.getSelectionModel().getSelectedItem().getReturnDate();
            int borrowID = borrowEquipments.getBorrowID();

            ZoneId defaultZoneId = ZoneId.systemDefault();
            Date returnedDate =
Date.from(LocalDateTime.now().toInstant(ZoneOffset.UTC));

            equipdettable_fx.getSelectionModel().getSelectedItem();



        String status = "Returned";
```

```
        Equipments eq =
equipmentPresenter.findEquipmentsById(equipmentId).get(0);

         List<ActiveUser> myUsername =
activeUserPresenter.getMyUsername();

        BorrowedEquipments selectedEquipment = new
BorrowedEquipments(borrowID, eq, myUsername.get(0), borrowDate,
returnDate, status, returnedDate);

        System.out.println(" the returned equipment is : "+
selectedEquipment);

        Boolean b =
this.borrowedEquipmentPresenter.returnEquipment(selectedEquipment);

equipdettable_fx.getItems().removeAll(equipdettable_fx.getSelectionModel(
).getSelectedItem());

         if(b==true){
            Date dateNow =
this.studentPresenter.getDateFromLocalDate(LocalDate.now());
            String notification = (myUsername.get(0).getUsername() + " "
+ dateNow +" - Returned an equipment named : "+ eq.getEquipment_name());
            System.out.println("notification is : "+ notification);
            this.studentPresenter.writeToFile(notification);

            Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
            alert.setTitle("Equipment Returned");
            alert.setHeaderText("Equipment "+eq.getEquipment_name()+" has
been Returned.");
            alert.showAndWait();


        }else{
            Date dateNow =
this.studentPresenter.getDateFromLocalDate(LocalDate.now());
            String notification = (myUsername.get(0).getUsername() + " "
+ dateNow +" - Failed to return an equipment named : "+
eq.getEquipment_name());
            System.out.println(notification);
            this.studentPresenter.writeToFile(notification);

            Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
            alert.setTitle("Failed to return Equipment.");
            alert.setHeaderText("Failed to return "+
eq.getEquipment_name());
            alert.showAndWait();
         }
        }
    }

    private void getAllBorrowedEquipmentList(List<BorrowedEquipments>
borrowedEquipmentList) {
        borrowId_column.setCellValueFactory(new
PropertyValueFactory<>("borrowID"));
```

```
        id_column.setCellValueFactory(new
PropertyValueFactory<>("equipmentID"));
        name_column.setCellValueFactory(new
PropertyValueFactory<>("equipmentName"));
        brand_column.setCellValueFactory(new
PropertyValueFactory<>("equipmentBrand"));
        type_column.setCellValueFactory(new
PropertyValueFactory<>("equipmentType"));
        borrowDate_column.setCellValueFactory(new
PropertyValueFactory<>("borrowDate"));
        returnDate_column.setCellValueFactory(new
PropertyValueFactory<>("returnDate"));
        status_column.setCellValueFactory(new
PropertyValueFactory<>("status"));


         ObservableList<BorrowedEquipments> equipmentsList =
FXCollections.observableList((borrowedEquipmentList));


         equipdettable_fx.setItems(equipmentsList);
         if (equipmentsList.isEmpty()) {
            equipdettable_fx.setPlaceholder(new Label("No records
found!"));
         }
    }

@FXML
    private void onSettingsButtonClicked(ActionEvent event) throws
IOException {
        Parent root =
FXMLLoader.load(getClass().getResource("StudentProfile.fxml"));
        Scene scene = new Scene(root);
        Stage window = (Stage) ((Node)
event.getSource()).getScene().getWindow();
        window.setScene(scene);
        window.show();
    }

    @FXML
    private void onNotificationButtonClicked(ActionEvent event) throws
IOException {
        Parent root =
FXMLLoader.load(getClass().getResource("NotificationStudent.fxml"));
        Scene scene = new Scene(root);
        Stage window = (Stage) ((Node)
event.getSource()).getScene().getWindow();
        window.setScene(scene);
        window.show();
    }

    @FXML
    private void onSearchEquipmentButtonClicked(ActionEvent event) throws
IOException {
        Parent root =
FXMLLoader.load(getClass().getResource("SearchEquipment.fxml"));
        Scene scene = new Scene(root);
```

```
        Stage window = (Stage) ((Node)
event.getSource()).getScene().getWindow();
        window.setScene(scene);
        window.show();
    }

    @FXML
    private void onBookingButtonClicked(ActionEvent event) throws
IOException {
        Parent root =
FXMLLoader.load(getClass().getResource("StudentHistory.fxml"));
        Scene scene = new Scene(root);
        Stage window = (Stage) ((Node)
event.getSource()).getScene().getWindow();
        window.setScene(scene);
        window.show();
    }
}
```