**TECHNICAL DOCUMENT**

**TOPIC:** “Grade management application”

**Group members:** Madina Askarova, Aliya Dauletkeldi, Zeinep Kaliaskar

**Project Idea:** “Application for tracking student progress” — is an application designed to store and manage student grades. The main idea is to create a user-friendly tool that allows users to save credentials and track their academic performance in one place.

**The goal of the project:** Goal of the project is to provide a user-friendly interface for managing passwords, grades, and subjects, making it easier to track academic results. The application should also be easy to use for users with any level of technical training.

**Main Features**

**1. Login and Password Management**:

* **Buttons**:
  + **buttonRegister** — adds a new login and password.
  + **buttonGenerate** — generates a random password.
* **Table**:
  + **tableWidgetPasswords** — displays user logins and passwords. Table rows: "Nickname" and "Password".
* **Description**: The user can input a login and password, which are then automatically added to the table. There is also a button to generate a random password for security.

**2. Grade and Subject Management**:

* **Buttons**:
  + **addGradeButton** — adds a grade for a specific subject.
  + **buttonDelete** — deletes the selected record from the table.
  + **buttonEdit** — edits the selected record in the table.
* **Table**:
  + **tableWidgetGrades** — displays the list of subjects and grades. Table rows: "Subject" and "Grade".
* **Description**: When a grade is entered in the **lineEditGrade** field and the **addGradeButton** is clicked, the application prompts the user to enter the subject name and adds this information to the table. The user can edit or delete records.

**3. Average Grade**:

* **Button**:
  + **labelAverage** — displays the average grade of all entered grades.
* **Description**: The application automatically calculates and displays the average grade based on all entered grades.

**Expected Functionality**

1. **Creating an Adaptive UI/UX**: The application should feature an intuitive and adaptive interface, making it easy to input and view data such as logins, passwords, grades, and subjects. Each control element (input fields, tables, buttons) should be clearly labeled and functional.
2. **Handling Invalid User Inputs**: The system must validate user input to ensure data is correct:
   * Grades should be in the range of 0 to 100.
   * Logins and passwords should not be empty.
   * Error messages should be shown when inputs are invalid, guiding the user to fix the issue.
3. **Handling Changes in Tables**: After the user adds or edits data in the tables (for passwords or grades), the application should immediately update and display the changes on the screen.
4. **Handling Dynamic Data**: After adding a new grade or password, the information should appear instantly in the corresponding table. The system should be quick and responsive to user input.

**Technical Implementation Details**

**Components Used**:

1. **Qt Framework**:
   * Qt is used for creating the graphical interface of the application. It provides powerful tools for implementing windowed applications with graphical interfaces.
2. **QTableWidget**:
   * For displaying tables with data about logins/passwords and grades/subjects. This component allows rows and columns of data to be displayed and edited.
3. **QInputDialog**:
   * For collecting data through dialog boxes, such as when requesting the name of the subject when adding or editing a grade.
4. **QPushButton**:
   * For creating buttons that perform actions like adding, editing, or deleting records.
5. **QLineEdit**:
   * Input fields for entering data, such as logins, passwords, and grades.

**How the Data Entry System Works**

1. **Login and Password Input**:
   * The user enters the login and password in the **lineEditUsername** and **lineEditPassword** fields.
   * After clicking the **buttonRegister**, the entered data is added to the **tableWidgetPasswords**.
2. **Password Generation**:
   * If needed, the user can generate a random password by clicking the **buttonGenerate**.
3. **Grade and Subject Input**:
   * The user enters a grade in the **lineEditGrade** field.
   * After clicking the **addGradeButton**, the application prompts the user to enter the subject name via a dialog box. The subject name and grade are then added to the **tableWidgetGrades**.
4. **Editing and Deleting Data**:
   * To edit or delete data, the user selects a row in the table and clicks the respective button: **buttonDelete** (for deleting) or **buttonEdit** (for editing).
5. **Average Grade Calculation and Display**:
   * The average grade of all entered grades is automatically calculated and displayed on the **labelAverage**.

**Expected Improvements**

1. **Saving Data to File**:
   * In the future, the application can be enhanced to save data (logins, passwords, grades, and subjects) to a file, so the information persists even after closing the program.
2. **Integration with Cloud Services**:
   * For added convenience, the application could be integrated with cloud storage services, enabling the user to synchronize data across multiple devices.
3. **Multi-user Support**:
   * In the future, the application could be adapted for use by multiple users. This would open up new opportunities for using the application in educational or workgroup environments.

**Conclusion**

The **"Password and Grade Manager"** application is a powerful tool for managing personal data (logins and passwords) and tracking academic performance. With this tool, users can securely store their data and conveniently track their grades. Future updates could add additional features, such as cloud synchronization or multi-user support, to make the application even more versatile.