A New System for designing a 'Student Aide' Application

[1]Ira Nath, [2]Tarpan Das, [3]Sumit Kumar Bera,
[4]Tanmoy Nath, [5]Vishal Kashyap, [6]Yachna Raj

[1,2,3,4,5,6]JIS College of Engineering, Kalyani

[1]Ira.nath@gmail.com, [2]tarpandas1@gmail.com, [3]berasumit956@gmail.com,
[4]nathtanmoy011@gmail.com, [5]vk41653@gmail.com, [6]yachnaraj2511@gmail.com

Abstract

Due to the current scenario across the continents, the premises of all the schools and colleges have been shut down to stop any kind of mass gatherings and, thus, avoid any unnecessary issues. To compensate this loss classes have been taking place online via various platforms. Students and guides check different online platforms in order to carry out various tasks, making the system a bit uneasy to work with. The very reason to systematize the whole process enabled us to produce a Computer application. The computer applications can be used to study contents, solve questions based on those contents and attend to or start online classes via a web browser. These are achieved by the use of JAVA Swing, which is a JAVA feature to produce desktop applications by using various GUI components, and NetBeans, an application tool, which provides the environment to work with these components.

Keywords: Student Aide, Online, JAVA Swing, NetBeans, etc.

1. Introduction

The current worldly issue has not been allowing on premise classrooms at the Educational Institutions. Thus, to overcome this, which is one of the many major issues caused by the pandemic, our team decided to lend a helping hand by helping normal education continue. As a result, we proceeded with implementing our idea into reality by developing a desktop application using JAVA Swing [1][2][3] and NetBeans [5].
Features of the app:

Contents based on syllabus,

MCQs based on chapters of the syllabus,

Easy access to classroom meetings via 'Zoom' and 'Google Meet'.

JAVA Swing: JAVA Swing is a part of JAVA Foundation Classes that can be used to make Windows Application. It is a lightweight and platform independent GUI toolkit, that uses buttons, scrollbars, text areas, etc for creating an windows application. It is built on the basis of AWT API and is completely written in JAVA.

Some key points used in the above definition:

- JAVA Foundation Class: Set of GUI components which simplify the development of desktop applications.
- MVC:

– The Model-View-Controller (MVC) is a architectural sequence that divides an application into three main components, namely, the model, the view, and the controller.
– The components are built to handle different developmental feature of an application.
– MVC is one of the more popularly used web development frameworks to create scalable and extensible projects at an industrial level.

- Pluggable feel and look: Changing the feel and look of a GUI during runtime.

The proposed work, simulation and result, future work and conclusion are discussed in section 2, 3, 4 and 5 respectively.

1.1 Objective

We needed to prepare a prototype for the development of a portal in the form of a PC application, which would provide study contents, quiz questions and an interface to start and/or access online classes via a web browser.

2. Proposed Work

Fig.1 shows the flowchart of the proposed work. In this app after login page, four modules are present. Those are CONTENTS, MCQS, VIDEO CONFERENCE and HELP. In the content section, all the contents are present chapter wise for all subjects present in a particular class. Students can also solve MCQS chapter wise of all subjects present in their syllabus. They can attend video conferencing through either Zoom or Google meet to clear their doubts.

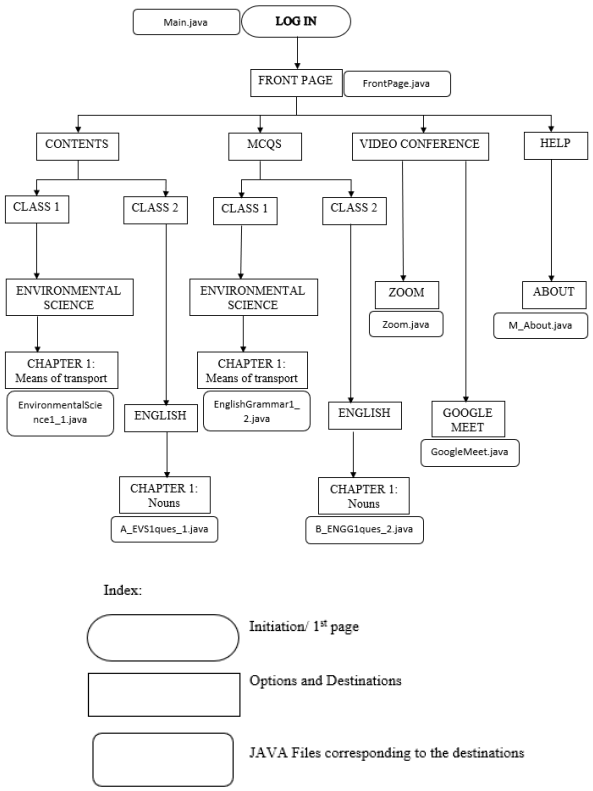


Figure 1: Proposed model for designing of a 'Student Aide' application

3. Simulation and Result

For the development of this project, the team used **JAVA Swing** and **NetBeans for Windows**. NetBeans is a free, open-source integrated development environment, licenced under 'Apache', which is used for developing applications using JAVA. It supports the development of all types of JAVA applications and provides high level of modularity. Further, for the development of this application, jdk 1.8.0 has been used on a Windows 10 device which provides all the necessary environment for the development of this project.

As the required simulations were made, the following results were obtained:

1. This is the first page to open while running the program. This is the typical login page. The **JTextField** [1][7] takes the Username as a text input and **JPasswordField** [1][7] takes the password as a text input. The Password Field is generally used to hide a text input. If the username and password to a record matches, the application Front page opens. We also introduced the concept of **JLabel** and **JButton** [1][7] for writing irreplaceable texts/labels and submitting a response respectively. Following is the interface involved. [Figure 2].

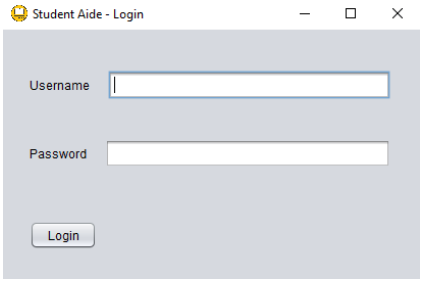


Figure 2: The Login Page

2. The Front Page or the main page is the second page added to this application. It contains links to various important pages. This is simplified by adding the **JMenuBar** [1][7] and multiple **JMenu(s)** [1][7] under which there are respective **JMenuItem(s)** [1][7]**.** The menu bar contains these Menu options: File, Contents, MCQs, Video Conferences and Help [Figure 3].

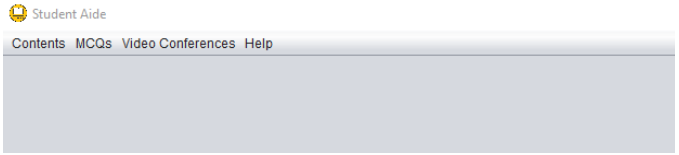
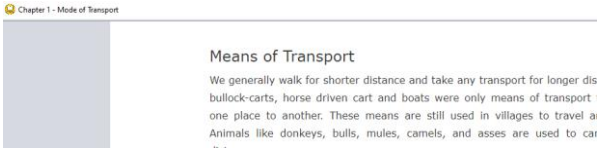
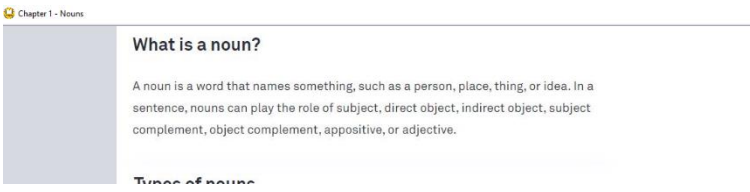


Figure 3: The Front Page

3. The Contents menu option provides links to content pages, where we used **JScrollPane** [1][7], which has what the students are to study. We also added Pictures to the Scroll Panel. Below are two simulation results justifying the statement [Figure 4].



(a)

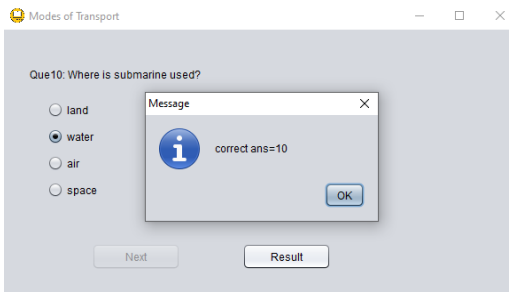


(b)

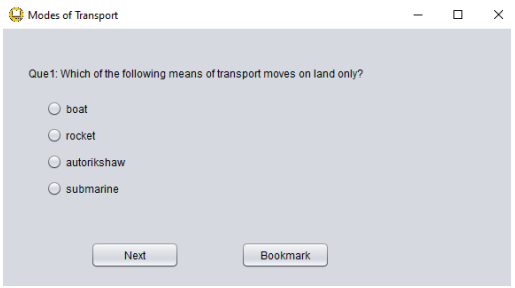
Figure 4: The Jframes (Chapters) opened using the JMenuElments under the JMenu option 'Contents'

4. The MCQs menu opens a quiz portal [6] respective to each Chapters under the content

Menu as mentioned earlier. We reused the concept of **JButtons** [1][7] and introduced the concept of **JRadioButtons** [1][7]. Here are the instances [Fig. 5 & 6].

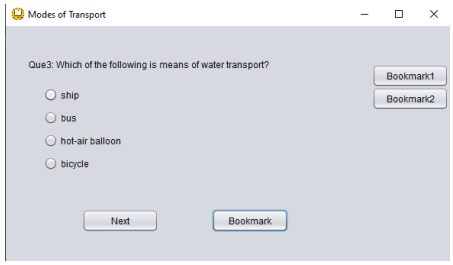


(a)

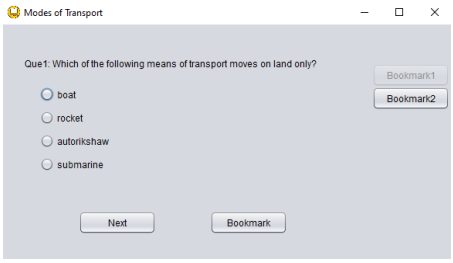


(b)

Figure 5: (a) & (b): Multiple instances of the working of JFrames containing the portions of MCQs



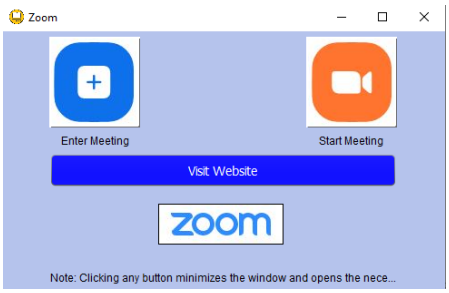
(a)



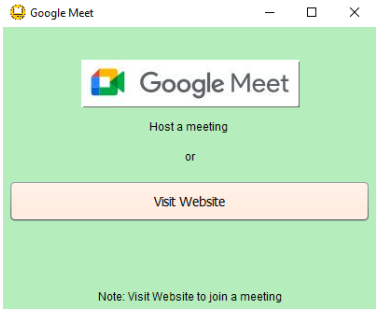
(b)

Figure 6: (a) Selecting the 'Bookmark' button bookmarks the link to the question selected, (b) Selected one of the bookmark buttons; leads to the respective question bookmarked.

5. The Video Conferences include two roughly designed **JFrames**[1][7] which contains **JButtons**[1][7] linking to websites [Figure 7].



(a)



(b)

Figure 7: JFrame(s) linking to the website for video conferences via (a) Zoom and (b) Google Meet

6. The 'Help' Menu option loads to a page 'About' which simply uses **JLabels**[1][7] for texts, pictures and links of actual contacts as shown in Figure 8.

Figure 8: The page with the names of contributes; uses JLabels for every name and picture.

4. Future Works

Well any application can be improved further to meet the requirements of the everchanging target audience. As our application provides a minimal application for a starter project, so many wonderful facets can be added to the application. Here are few ideas to work upon.

- Making the application online: While the application is mainly to provide support students offline, and also part of it is already implemented to work online (Video Conferences), extra links to study materials and worksheets to dedicated websites, and many features can be added to make the application work online and increase its prowess.
- Making it suitable for teachers/professors: We can add so many teachers for teachers, like adding students' names to the attendance sheet via connecting it to a database, inserting new worksheets, materials, etc. However, this needs features like the AWS or any other reliable cloud providers to be continued, as it might consume a lot of memory.
- Secured login via a database: The project has a login page, however it has a dedicated username and password and no signup page, since no database is added. A normal application wouldn't work like this, and it is the only feature in the database which is not independent on its own. The use of database will fix this issue in no time.
- Improving interfaces: The application is meant to appeal to school kids. As a result of this the interface can be improved.

5. Conclusion

Thus, we created an application using JAVA Swing and its features. The application has multiple Frames linked to one another and works well to provide a basic interface for a student at home to attend to classes and get the required study help needed, all while login in to his/her Personal Computer. This application also holds the potential to be adopted to schools. Teachers/Professors can distribute the application among its students to teach them the required materials needed via online classes and thus, provide an unparalleled studying experience.

References

[1]Singh, Chaitanya, and Laura says. "Java Swing Tutorial for Beginners." Beginnersbook.com, Beginnersbook, 5 July 2015, beginnersbook.com/2015/07/java-swing-tutorial/

[2]Eckstein, Robert & Loy, Marc & Wood, Dave. (1998). Java Swing

[3] J. D. Newmarch, "Testing Java Swing-based applications," Proceedings Technology of Object-Oriented Languages and Systems (Cat. No.PR00393), Nanjing, 1999, pp. 156-165, doi: 10.1109/TOOLS.1999.796479

[4] Mukhtar, M.I., and Galadanci, B. S. "THE DESIGN AND DEVELOPMENT OF A LEARNING TOOL FOR DEMONSTRATING AUTOMATIC JAVA CODE GENERATION FROM UML CLASS DIAGRAMS", Dutse Journal of Pure and Applied Sciences 2(1), June 2016

[5] Magdalena Dukielska, Sroka "JavaSpaces NetBeans — a Linda Workbench for Distributed Programming Course", ITiCSE '10: Proceedings of the fifteenth annual conference on Innovation and technology in computer science education, June, 2010

[6]Gordon, E. O. & Malloy, T. E. (2002). Online Homework/Quiz/Exam Applet: Freely available Java software for evaluating performance online. Behavior Research Methods Instruments & Computers, 34, 241-244

[7]J. Jeon, X. Qiu, J. Fetter-Degges, J. S. Foster and A. Solar-Lezama, "Synthesizing Framework Models for Symbolic Execution," 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE), Austin, TX, 2016, pp. 156-167, doi: 10.1145/2884781.2884856.

Appendix

The common methods and instructions used are as follows:

- setSize(): Set size to JFrames, JPanels, etc.
- setDefaultCloseOperation(): For choosing closing JFrame. We generally used
- add(): Adding something to something else.
- setVisible(): To make something visible.
- getText(): Take input from a textfield. (e.g. Username.getText(), where username is the class).
- setIconImage(Toolkit.getDefaultToolkit().getImage(getClass().getResource("Project_Icon.png"))) : To set the image 'Project_Icon.png' as the application icon.
- getVerticalScrollBar().setUnitIncrement(int value) : Increment the scroll speed.
- JRadioButton: Radio buttons are a group of buttons, from where only one button can be chosen at a time
- Desktop.getDesktop().browse(new URL("website").toURI()): to link a website.

Codes for quiz portal:

```
if(e.getSource()==b1){  //b1 button is for submitting answer

        if(check())

           count=count+1;

        current++; set();
```

```
        if(current==9){
           b1.setEnabled(false);
           b2.setText("Result");  //b2 button is for choosing next question
        } }
      if(e.getActionCommand().equals("Bookmark")){ //Action on pressing “Bookmark” button
        JButton bk=new JButton("Bookmark"+x);
        bk.setBounds(480,20+30*x,100,30);  add(bk);
        bk.addActionListener(this);  m[x]=current;
        x++; current++; set();
        if(current==9)
           b2.setText("Result");
        setVisible(false);  setVisible(true);
      }
      for(int i=0,y=1;i<x;i++,y++)  {
       if(e.getActionCommand().equals("Bookmark"+y))  {
          if(check())
             count=count+1;
          now=current;  current=m[y];  set();
          ((JButton)e.getSource()).setEnabled(false);
          current=now;
       } }
      if(e.getActionCommand().equals("Result")) { //Displaying Final Result
          if(check())
             count=count+1;
          current++;
          JOptionPane.showMessageDialog(this,"correct ans="+count);
          this.dispose();
```