**3. You should have a folder named design containing**

**(a) High-level design document (in HTML).**

**i. Containing one or more UML class diagrams showing your packages and the**

**dependence relations between these packages. Also show library packages**

**or third-party packages that are used.**

**ii. For each of your packages, a short description of the purpose of the package.**

The implementation of the project is done with the MVC software design pattern. Hence the main packages used are as follows:  
  
1) Model Package:

The model package is responsible to update the view package after it receives the manipulations from the controller package. The model package is the central component of this pattern. It is responsible for directly managing the data application data, logic, and rules of the application.

Contains: exit.java, fontSize.java, main.java, panelInfo.java etc.

2) View Package:

Any representation of the application and information is handled with the help of this package. Multiple different customizations according to our requirements are done so that we can create a GUI based on it which serves our purpose.

Contains: addpanel.java, ComponentMover.java, ComponentResizer.java, FileFilter.java, JpanelSlider.java etc.

3) Controller Package:

The purpose of the controller package is to accept the input and convert it to use for the model and the view packages. It performs interactions on the data model objects. It receives the input, validates it optionally and then proceeds to pass it to the model.

Contains: FontChooser.java, Jtext.java, MainFrame.java, etc.

**iii. For any third-party packages, you use, explain the purpose of the package**

**and where its documentation can be found.**

We have created the application interface GUI using the JTattoo, which is used for developing GUI with the help of swing applications. It enhances the look in comparison to the standard JDK. The documentation can be found at the following web address along with installation and usage procedure:

http://www.jtattoo.net/index.html

**iv. For each of your packages, one or more class diagrams showing the classes**

**and interfaces inside the package, associations between them and associations with classes in other packages that are depended on.**

**v. For each class and interface, a description of the purpose of the class. I.e.,**

**what it represents or does.**

**(b) A test plan that explains how you intend to test each package and class.**

**(c) A short document that explains how to compile and execute your code.**

**4. You should have a folder named src (or source) that contains your source code.**

**(a) Each class should be documented using JavaDoc (for Java), TSDoc (for Typescript), or Dox (for Haxe). Each public method should also be documented.**

|  |  |
| --- | --- |
| **Class Name** | **Functionality** |
| **Main** | It’s the main class from where our code is running. |
| **MainFrame** | It’s the GUI of whole app that’s shows buttons (File, Home, View, Edit, Camera, Presentation, Transitions), attribute panel and panel (left slides panel and the main slide panel). |
| **JPanelSlider** | It swaps the main slide to left and right using media buttons |
| **FileFilter** | Its functionality is to open, save , saves and exit the file. |
| **Addpanel** | It actually add the slides by clicking the Add Slide button(show in Home Panel). |
| **ComponentResizer** | The main functionality of this class is to resize the components which we add on the slides like shapes and text boxes. |
| **ComponentMover** | It’s functionality is to move the components on the slide anywhere in x-y direction. |
| **FontSize** | We change the font-size, color, and its font-family by using this class. |
| **PanelInfo** | It stores all the info about the panels we add into it. |
| **JText** | This class helps to add text on the slides. |

**5. You should have a folder named tstsrc (or test-source) that contains your automated**

**tests. For Java, please use JUnit 5 (or 4). For TypeScript, I recommend using Mocha.**

**For Haxe, I’ve used Buddy with success. Your model, and controller should be tested.**

**Also any classes in the view that it makes sense to test should be tested.**