

# **Six Weeks Industrial Training Project Report**

On

**Typing Guru (A Typing Tutor)**

Submitted in the Partial fulfillment of the requirement for the Award of Degree of

**Bachelor of Technology**

In

**INFORMANTION TECHNOLOGY**

**Batch**

**2017-2021**



**Submitted to**

Er. Vinod Sharma (HOD)

Department of CSE & IT

**Submitted by**

Abdhesh Nayak

B.Tech IT 5<sup>th</sup> Sem (1701448)

## ACKNOWLEDGEMENT

A study or a project of this volume can never be the outcome of a single person or just a mere group of dedicated students, so we express our profound sense of gratitude to those who extended their whole hearted help and support to us in completing our project because successful completion of any work requires guidance and help from a number of people .Firstly, it gives us immense pleasure to acknowledge our institute **AMRITSAR COLLEGE OF ENGINEERING AND TECHNOLOGY** for providing us an opportunity in developing a project on “**Typing Guru**”.

It would not have been possible to see through the undertaken project without the guidance of **Udemy Online Course**. It was purely on the basis of their experience and knowledge that we able to clear all the theoretical and technical hurdles during the development phases of this project work.

## Index

S.N.	Contents	Page Number
1	Training Objective	1
2	Organization Profile	2
3	Introduction of training course	3-4
4	Introduction to the Project	4-6
5	Syllabus Contains of this Course	6-7
6	Source Code	8
7	Snapshots	9-99

## **Training Objective**

An immediate objective of training is to give employees the skill they need to become better workers, resulting in financial gain. Other objectives include creating a supportive workplace. So employees know they are valued and feel more satisfaction in their jobs. Today all organizations are looking for those graduate /degree holders who are technically sound, creative and analytical. They don't want to spend time and money for training before putting employees on the job. That's the main problem faced by fresh graduates who apply for a job. A quality solution for this problem is to take industrial exposure very seriously and gain practical knowledge as much as you can do. This way both employer and employee will be benefited up to some extent.

- Helps to gain in-depth technical knowledge of opted engineering stream
- Helps to enhance technical skills in real time environment
- Helps to understand the area of interest and selection of an area of specialization
- Helps to learn basics of how to work as a team member to complete given tasks
- Helps to improve awareness of the industrial environment and work culture of the specific industry
- Real-time work and workshop projects help to learn more analytically
- Interaction with experts help to solve queries with practical exposure
- Certificate obtained from reputed organization give weightage to resume or curriculum vitae
- Project during summer training helps to judge trainee's capabilities and skills

Obtain highest marks in theory and practical exams is played a vital role in most of the case, but technical skills and aptitude learned in summer training helps to acquire a dream job.

## Organization Profile

The Training where we are pursuing our six Weeks Training from Udemmy Online Training Center.

### DETAILS OF UDEMY ONLINE TRAINING CENTER

**Udemmy** is an online learning and teaching marketplace with over 100000 courses and 24 million students. Udemmy is the leading global marketplace for teaching and learning, connecting students everywhere to the world's best instruction anywhere.

Udemmy's mission is to improve lives through learning. Our global marketplace features an extensive, multi-language library, which includes over 130,000 courses taught by expert instructors. You can take courses across a wide range of categories, some of which include: business & entrepreneurship, programming, academics, the arts, health & fitness, language, music, technology, games, and more. With instructors adding new content every day, be sure to check back regularly for the latest courses. The opportunities for learning are endless!

### UDEMY FEATURES

- Teach Hub
- Training Videos
- Earn Extra Income
- External Partnership Promotions
- Course Catalog
- Coupons and Discounts
- Search and Discovery
- Course Quality Checklist
- Course Marketing

#### Device Supported

- Web – based
- Ios
- Android
- Desktop

#### Customers type

- Small business
- Medium business
- Enterprise

#### Customer Support Types

- Phone
- Online

# INTRODUCTION OF TRAINING COURSE

## Java Programming Masterclasses

**Java** is an object-oriented, cross platform, multi-purpose programming language produced by Sun Microsystems. First released in 1995, it was developed to be a machine independent web technology. It was based on C and C++ syntax to make it easy for programmers from those communities to learn. Since then, it has earned a prominent place in the world of computer programming.

Java has many characteristics that have contributed to its popularity:

- **Platform independence** - Many languages are compatible with only one platform. Java was specifically designed so that it would run on any computer, regardless if it was running Windows, Linux, Mac, Unix or any of the other operating systems.
- **Simple and easy to use** - Java's creators tried to design it so code could be written efficiently and easily.
- **Multi-functional** - Java can produce many applications from command-line programs to applets to Swing windows (basically, sophisticated graphical user interfaces).

Java does have some drawbacks. Since it has automated garbage collection, it can tend to use more memory than other similar languages. There are often implementation differences on different platforms, which have led to Java being described as a "write once, test everywhere" system. Lastly, since it uses an abstract "virtual machine", a generic Java program doesn't have access to the Native API's on a system directly. None of these issues are fatal, but it can mean that Java isn't an appropriate choice for a particular piece of software.

### Features of Java Programming

- |                        |                        |
|------------------------|------------------------|
| ➤ Simple               | ➤ Architecture neutral |
| ➤ Object-Oriented      | ➤ Interpreted          |
| ➤ Portable             | ➤ High Performance     |
| ➤ Platform independent | ➤ Multithreaded        |
| ➤ Secured              | ➤ Distributed          |
| ➤ Robust               | ➤ Dynamic              |

### Syllabus Contains of this Course

- All the essential Java keywords, operators, statements, and expressions needed to fully understand exactly what you're coding and why - making programming easy to grasp and less frustrating

- You will learn the answers to questions like What is a Java class, what is polymorphism and inheritance and to apply them to your java apps.
- How to safely download and install all necessary coding tools with less time and no frustrating installations or setups
- Complete chapters on object-oriented programming and many aspects of the Java API (the protocols and tools for building applications) so you can code for all platforms and derestrict your program's user base (and potential sales)
- How to develop powerful Java applications using one of the most powerful Integrated Development Environments on the market, IntelliJ IDEA! - Meaning you can code functional programs easier. **IntelliJ** has both a **FREE** and PAID version, and you can use either in this course.

## Introduction to the Project

### Typing Guru

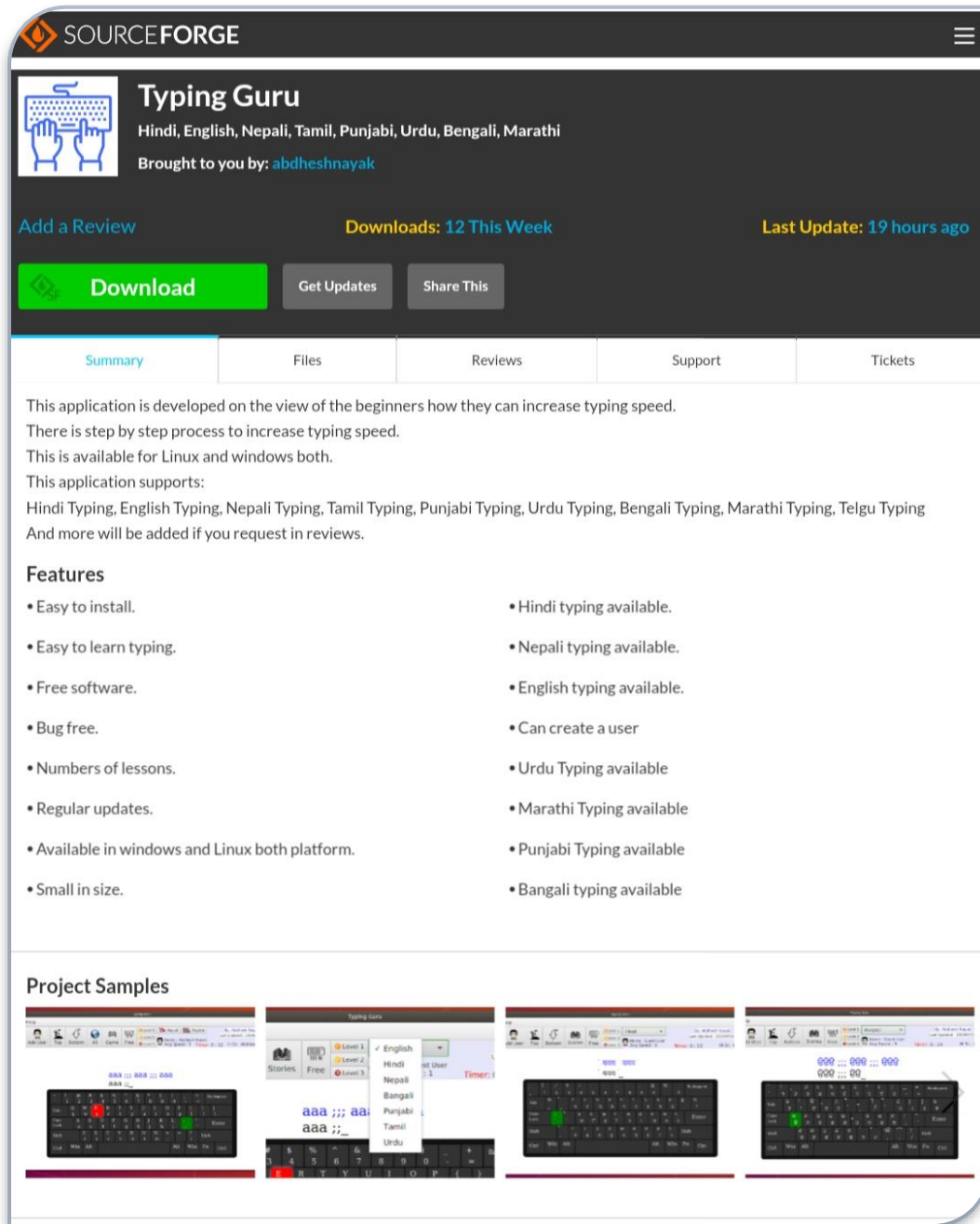
Typing Guru is an application that enables users to learn Typing For a beginner in a very fast way. This application is helpful to the department of the organization which provides basic Computer Training or any individual who wants to learn Typing in a very fast manner.

It is simple to understand and can be used by anyone and on any platform like Windows, Linux, Mac-OS. A user who is not even can put their finger on the keyboard can start their journey on this application. It is user-friendly and just asks the user to follow step by step operations by giving him a few options. It is fast and can be used to learn to type in several languages keyboard.

This application is developed for use on multiple platforms because we know all java is platform-independent, we can use it on various platforms. Application Is bundled as 'MSI' installer for windows which include the runtime environment that is Java Runtime Environment (JRE), Graphics Files, etc.

## How Can User Get the Program?

This project is uploaded on GitHub (Source Code) and Source Forge (file that can be used by user).



GitHub Link: <https://github.com/abdhesh-nayak/TypingGuru>

Source Forge Link: <https://sourceforge.net/projects/typingguru/>



## Difficulties in Development

Problem: After The completion of my project when tried to export it for the users it shows me jdk11 not supports fxdeploy.

Solution: The problem was I used jdk11 which comes without javafx bundle so I learned it from google and changed the jdk11 to jdk1.8 then my problem is solved.

## SOFTWARE AND HARDWARE REQUIREMENT

### SOFTWARE

#### IntelliJ Idea

IntelliJ is one of the most powerful and popular Integrated Development Environments (IDE) for Java. It is developed and maintained by **Jet Brains** and available as community and ultimate edition. This feature rich IDE enables rapid development and helps in improving code quality

IDE stands for Integrated Development Environment. It is a combination of multiple tools, which make software development process easier, robust and less error-prone. It has following benefits over plain text editor –

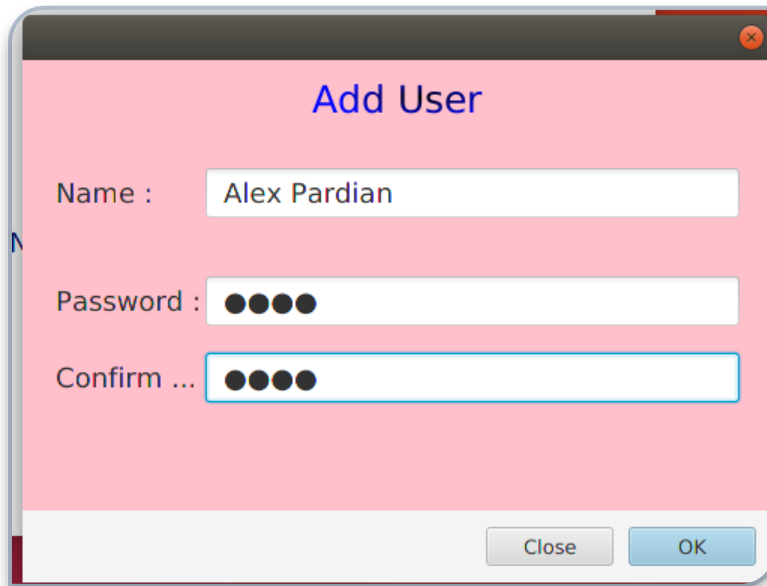
- Integration with useful tools like compiler, debugger, version control system, build tools, various frameworks, application profilers and so on.
- Supports code navigation, code completion, code refactoring and code generation features which boosts development process.
- Supports unit testing, integration testing and code coverage via plug-ins.
- Provides rich set of plug-ins to enhance **IDE** functionality further.

### HARDWARE

#### Laptop

A laptop computer is a portable personal computer powered by a battery, or an AC cord plugged into an electrical outlet, which is also used to charge the battery. Laptops have an attached keyboard and a touchpad, trackball, or isometric joystick used for navigation. A laptop also has a thin display screen that is attached and can be folded flat for transport.

## Snapshots:



A dialog box titled "Add User" with a pink background. It contains three input fields: "Name" with the text "Alex Pardian", "Password" with four black dots, and "Confirm ..." with four black dots. At the bottom right are "Close" and "OK" buttons.

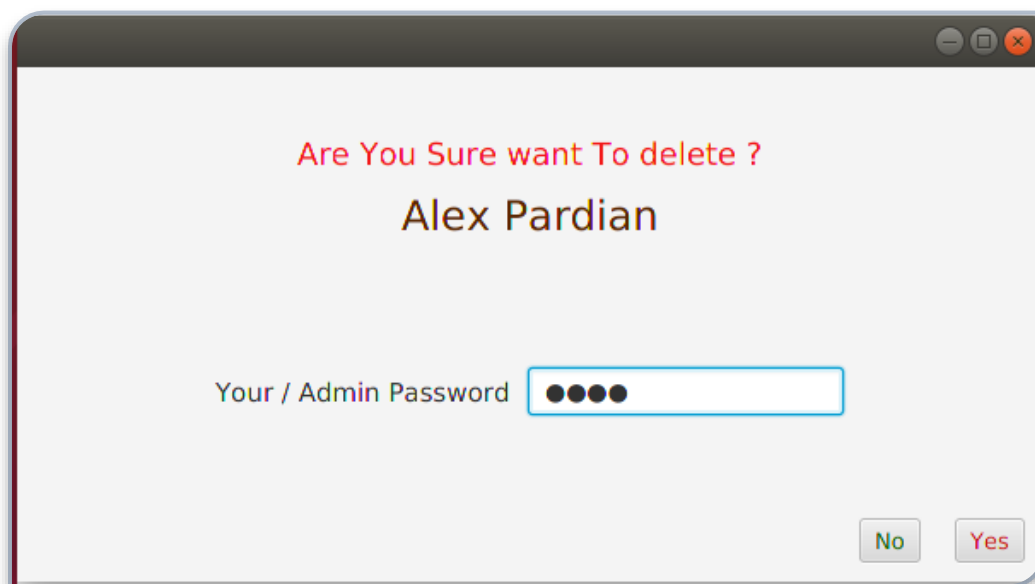
**Add User**

Name : Alex Pardian

Password : ●●●●

Confirm ... ●●●●

Close OK



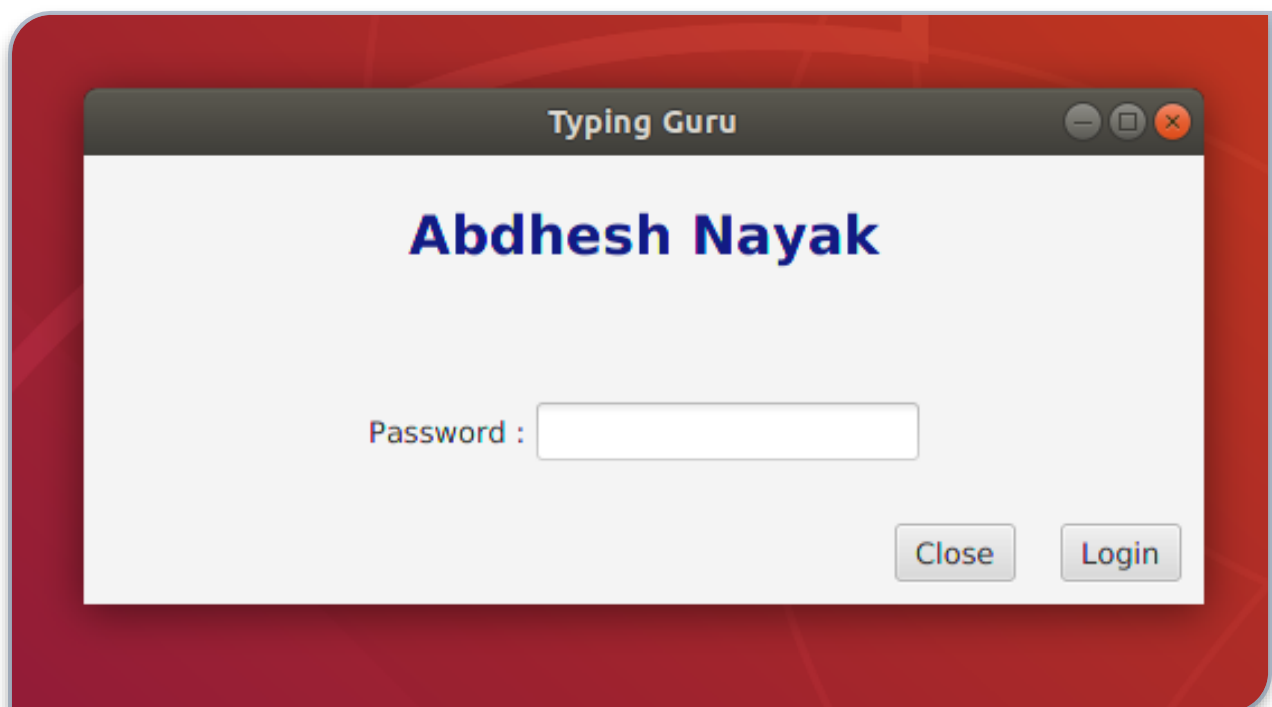
A dialog box with a light gray background. It displays the text "Are You Sure want To delete ?" in red, followed by "Alex Pardian" in black. Below is a label "Your / Admin Password" and a password input field with four black dots. At the bottom right are "No" and "Yes" buttons.

Are You Sure want To delete ?

Alex Pardian

Your / Admin Password ●●●●

No Yes



Typing Guru

File Help

Home Add User Top Bottom Stories Free

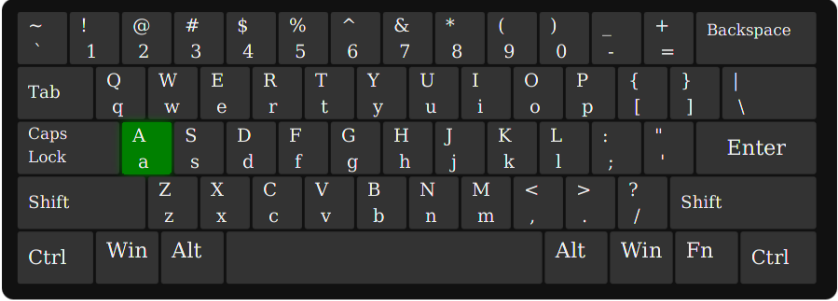
Level 1 English  
Level 2  
Level 3

By: Abdhesh Nayak  
Last Updated : 2019/07/14

Name : Guest User  
Avg Speed : 24

Timer: 0 : 5 Hi Sc: Guest User 0

aaa ;;; aaa ;;; aaa  
aaa ;;; aa\_



Typing Guru

File Help

Home Add User Top Bottom Stories Free

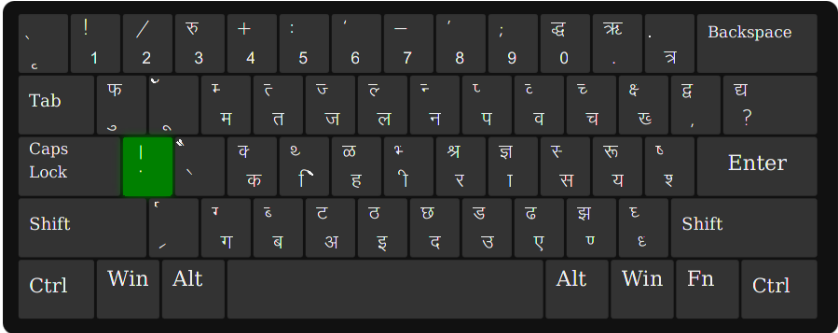
Level 1 Hindi  
Level 2  
Level 3

By: Abdhesh Nayak  
Last Updated : 2019/07/14

Name : Guest User  
Avg Speed : 9

Timer: 0 : 13 Hi Sc: Guest User 0

ययय ययय  
ययय \_



Typing Guru

File Help

Home Add User Top Bottom Stories Free

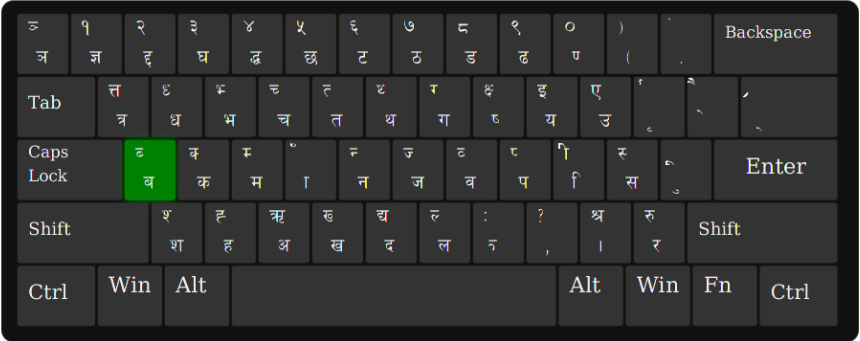
Level 1 Nepali  
Level 2  
Level 3

By: Abdhesh Nayak  
Last Updated : 2019/07/14

Name : Guest User  
Avg Speed : 6

Timer: 0 : 19 Hi Sc: Guest User 0

बबब ससस बबब ससस बबब  
बबब ससस बब



Typing Guru

File Help

Home Add User Top Bottom Stories Free

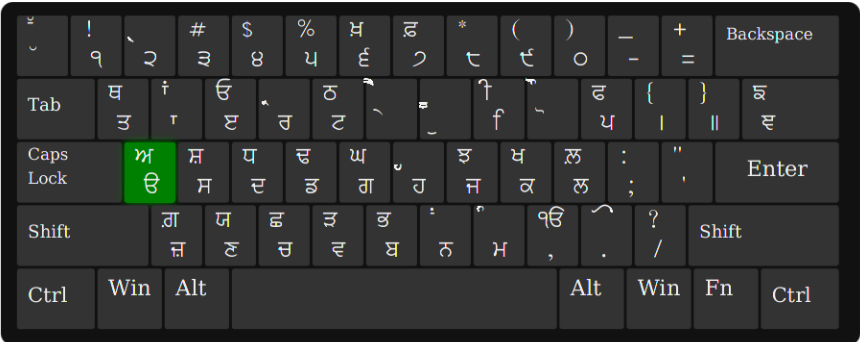
Level 1 Punjabi  
Level 2  
Level 3

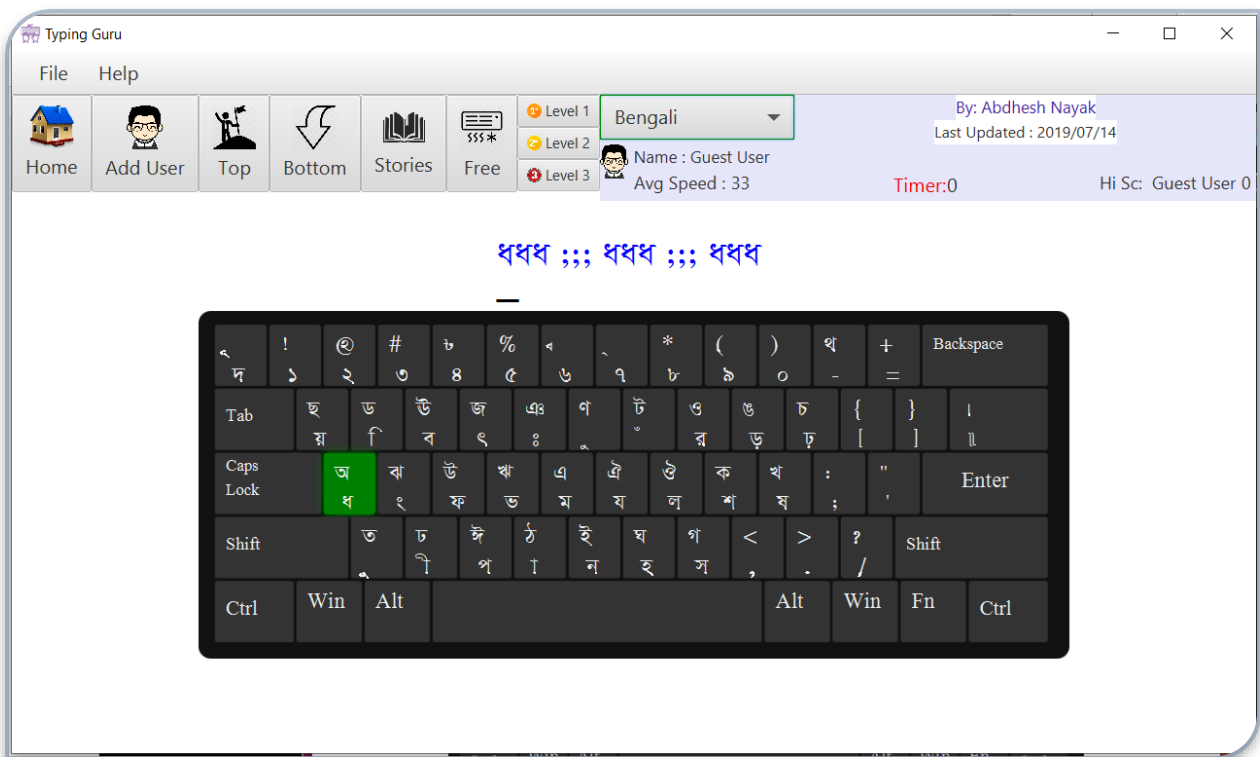
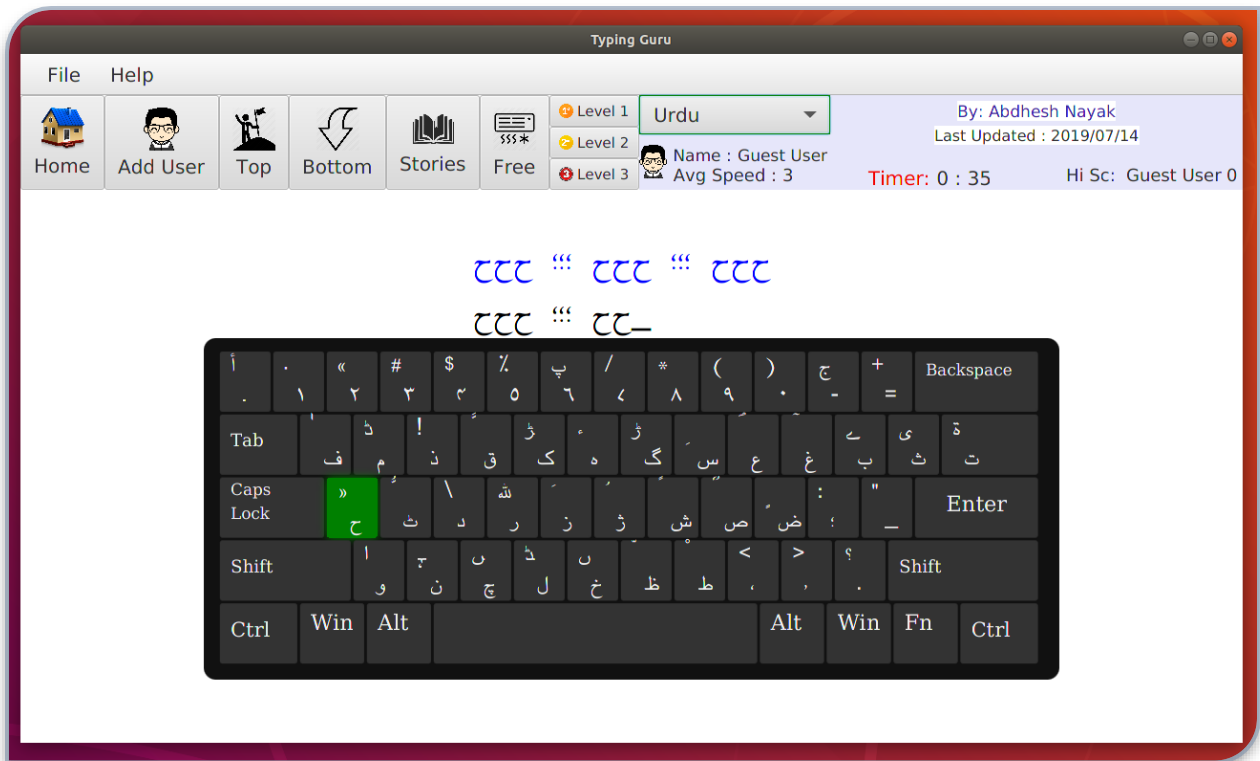
By: Abdhesh Nayak  
Last Updated : 2019/07/14

Name : Guest User  
Avg Speed : 5

Timer: 0 : 24 Hi Sc: Guest User 0

ੳੳੳ ;;; ੳੳੳ ;;; ੳੳੳ  
ੳੳੳ ;;; ੳੳ





Home

Add User

Top

Bottom

Stories

Free

Level 1

Level 2

Level 3

Tamil

By: Abdhesh Nayak

Last Updated : 2019/07/14

Name : Guest User

Avg Speed : 33

Timer:0

Hi Sc: Guest User 0

யயய' யயய' யயய

ஷ	!	ஶ	சூ	கூ	ழூ	ஶூ	ழூ	(	)	ஶ	ஶ	Backspace
ஶ	1	2	3	4	5	6	7	8	9	0	-	ஶ
Tab	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	'
Caps Lock	ய	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	Enter
Shift	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	ஶ	Shift
Ctrl	Win	Alt									Alt	Win

Abdhesh Nayak

## Free Typing

**When** one loves ones art no service seems too hard. That is our premise. This story shall draw a conclusion from it, and show at the same time that the premise is incorrect. That will be a new thing in logic, and a feat in story-telling somewhat older than the Great Wall of China. Joe Larrabee came out of the post-oak flats of the Middle West pulsing with a genius for pictorial art. At six he drew a

~	!	@	#	\$	%	^	&	*	(	)	-	+	Backspace
`	1	2	3	4	5	6	7	8	9	0	-	=	
Tab	Q	W	E	R	T	Y	U	I	O	P	{	}	
Caps Lock	A	S	D	F	G	H	J	K	L	:	"	'	Enter
Shift	Z	X	C	V	B	N	M	<	>	?	/	Shift	
Ctrl	Win	Alt									Alt	Win	Ctrl

## Source Code:

### Main.java

```
1  package com.abdheshnayak.typingguru;
2
3  import com.abdheshnayak.typingguru.JavaClasses.staticData;
4  import javafx.application.Application;
5  import javafx.fxml.FXMLLoader;
6  import javafx.scene.Parent;
7  import javafx.scene.Scene;
8  import javafx.scene.image.Image;
9  import javafx.stage.Stage;
10
11 import java.io.FileInputStream;
12
13 public class Main extends Application {
14
15     @Override
16     public void start(Stage primaryStage) throws Exception{
17         System.out.println("Program Started Succesfully");
18         //      staticData.usrname=a.usrdata.users.get()
19         //      Parent root =
FXMLLoader.load(getClass().getResource("FreeWriting.fxml"));
20         Parent root =
FXMLLoader.load(getClass().getResource("UserSelector.fxml"));
21         //      Parent root =
FXMLLoader.load(getClass().getResource("ExampleFXML.fxml"));
22         primaryStage.setTitle("Typing Guru");
23
24         FileInputStream input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/DialogIcon.png");
25
26         primaryStage.getIcons().add(new Image(input));
27         //      primaryStage.setScene(new Scene(root,1280,740));
28         primaryStage.setScene(new Scene(root, 600, 400));
29         primaryStage.show();
30     }
31
32
33     public static void main(String[] args) {
34         staticData.myDir=System.getProperty("user.dir");
35         //      System.out.println(staticData.myDir);
36         launch(args);
37     }
38 }
39
```

### MainWindowController.java

```
1  package com.abdheshnayak.typingguru;
2
```



```
3
4 import com.abdheshnayak.typingguru.JavaClasses.fileReaderClass;
5 import com.abdheshnayak.typingguru.JavaClasses.staticData;
6 import com.abdheshnayak.typingguru.JavaClasses.tempDatas;
7 import javafx.animation.Animation;
8 import javafx.animation.KeyFrame;
9 import javafx.animation.Timeline;
10 import javafx.event.ActionEvent;
11 import javafx.event.EventHandler;
12 import javafx.fxml.FXML;
13 import javafx.fxml.FXMLLoader;
14 import javafx.scene.Scene;
15 import javafx.scene.control.*;
16 import javafx.scene.image.Image;
17 import javafx.scene.image.ImageView;
18 import javafx.scene.input.KeyEvent;
19 import javafx.scene.input.MouseEvent;
20 import javafx.scene.layout.BorderPane;
21 import javafx.scene.layout.GridPane;
22 import javafx.scene.web.WebEngine;
23 import javafx.scene.web.WebView;
24 import javafx.stage.Stage;
25 import javafx.stage.WindowEvent;
26 import javafx.util.Duration;
27
28 import java.io.FileInputStream;
29 import java.io.FileNotFoundException;
30 import java.io.IOException;
31 import java.util.HashMap;
32 import java.util.Map;
33 import java.util.Optional;
34
35 import static com.abdheshnayak.typingguru.JavaClasses.SorterClass.ASC;
36 import static
com.abdheshnayak.typingguru.JavaClasses.SorterClass.sortByComparator;
37
38 public class MainWindowController {
39
40     public static Stage stage = new Stage();
41     @FXML
42     private MenuItem aboutAppbtn;
43
44     @FXML
45     private ImageView StoriesIcon;
46     @FXML
47     private Button btnStories;
48     @FXML
49     private ChoiceBox LanguageSelector;
50     @FXML
51     private MenuItem quite;
52     @FXML
53     private Label DevLevel;
54     @FXML
55     private Button btnFree;
56     @FXML
57     private Label highScoreLablename;
58     @FXML
```

```
59     private Label highScoreLable;
60     @FXML
61     private Label timerLabel;
62     @FXML
63     private Label speedLabel;
64     @FXML
65     private Button btnBottom;
66     @FXML
67     private Button btnTop;
68     @FXML
69     private Label userNameLabel;
70     @FXML
71     private GridPane MainGridPane;
72     @FXML
73     private ImageView adduserIcon;
74     @FXML
75     private Button btnAddUser;
76     @FXML
77     public WebView webopen;
78     @FXML
79     private ImageView userIcon;
80     @FXML
81     private ImageView level1Icon;
82     @FXML
83     private ImageView level2Icon;
84     @FXML
85     private ImageView level3Icon;
86     //     @FXML
87     //     private ImageView AllIcon;
88     //     @FXML
89     //     private ImageView GameIcon;
90     @FXML
91     private ImageView FreeIcon;
92     @FXML
93     private ImageView bottomIcon;
94     @FXML
95     private ImageView topicon;
96     @FXML
97     private ImageView HomeIcon;
98     //     @FXML
99     //     private ImageView npflag;
100    //     @FXML
101    //     private ImageView Enflag;
102    //     @FXML
103    //     private Label userinput1;
104    //     @FXML
105    //     private Label userinput2;
106    //     @FXML
107    //     private Label userinput3;
108    //     @FXML
109    //     private Label taskuser;
110    @FXML
111    private Button btnNepali;
112    @FXML
113    private Button btnEnglish;
114    private String task;
115    private String taskC;
```

```

116 //      @FXML
117 //      private Button btnHome;
118 //      @FXML
119 //      private Button btnAll;
120 //      @FXML
121 //      private Button btnGame;
122 //      @FXML
123 //      private Button btnFree;
124      @FXML
125      private Button level1;
126      @FXML
127      private Button level2;
128      @FXML
129      private Map<String, Integer> keys = new HashMap<>();
130      @FXML
131      private Button level3;
132      private String tempKeyboard;
133      private static tempDatas td=new tempDatas();
134
135
136      private void setImageFun() throws IOException {
137
138          FileInputStream input = new
139      FileInputStream(staticData.myDir+"/.src/Images/Icons/usericon.png");
140          Image image = new Image(input);
141          userIcon.setImage(image);
142
143      //          input = new
144      FileInputStream(staticData.myDir+"/.src/Images/Icons/nepali-flag.png");
145      //          image = new Image(input);
146      //          npflag.setImage(image);
147
148      input = new
149      FileInputStream(staticData.myDir+"/.src/Images/Icons/usericon.png");
150      image = new Image(input);
151      adduserIcon.setImage(image);
152
153      //          input = new
154      FileInputStream(staticData.myDir+"/.src/Images/Icons/english-flag.png");
155      //          image = new Image(input);
156      //          Enflag.setImage(image);
157
158      input = new
159      FileInputStream(staticData.myDir+"/.src/Images/Icons/home.png");
160      image = new Image(input);
161      HomeIcon.setImage(image);
162
163      input = new
164      FileInputStream(staticData.myDir+"/.src/Images/Icons/topIcon.png");
165      image = new Image(input);
166      topicon.setImage(image);
167
168      input = new
169      FileInputStream(staticData.myDir+"/.src/Images/Icons/Down.png");

```

```

166         image = new Image(input);
167         bottomIcon.setImage(image);
168
169         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/Stories.png");
170         image = new Image(input);
171         StoriesIcon.setImage(image);
172
173         //         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/game.png");
174         //         image = new Image(input);
175         //         GameIcon.setImage(image);
176
177         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/freeWriting.png");
178         image = new Image(input);
179         FreeIcon.setImage(image);
180
181         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/level1icon.png");
182         image = new Image(input);
183         level1Icon.setImage(image);
184
185
186         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/level2icon.png");
187         image = new Image(input);
188         level2Icon.setImage(image);
189
190         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/level3icon.png");
191         image = new Image(input);
192         level3Icon.setImage(image);
193         keyboardLoad();
194     }
195
196     private void keyboardLoad() {
197         WebEngine webEngine= webopen.getEngine();
198         webEngine.load(tempKeyboard);
199     }
200
201     public void QuiteAction(ActionEvent actionEvent) {
202
203         Stage s= (Stage) btnBottom.getScene().getWindow();
204         s.close();
205
206     }
207
208     public void LanguageController(ActionEvent actionEvent) throws
IOException, ClassNotFoundException {
209
210         if(actionEvent.getSource().equals(LanguageSelector)) {
211
212             staticData.unvLanguage=LanguageSelector.getValue().toString();
213             td.hw.ChangeLanguage(LanguageSelector.getValue().toString());
214             td.t.run();

```

```

214         keyboardLoad();
215     }
216
217 }
218
219 public void AboutAppFun(ActionEvent actionEvent) throws IOException
220 {
221     if (td.dialogptr) {
222         td.dialogptr = false;
223         FileInputStream input = new FileInputStream(staticData.myDir
224 + "/.src/Images/Icons/DialogIcon.png");
225         stage.getIcons().add(new Image(input));
226         // Stage stage = (Stage)
227 btnBottom.getScene().getWindow();
228         GridPane root;
229
230         root = (GridPane)
231 FXMLLoader.load(getClass().getResource("aboutApp.fxml"));
232         Scene scene = new Scene(root, 960, 540);
233         stage.setScene(scene);
234
235         stage.showAndWait();
236
237         td.dialogptr = true;
238     }
239 }
240
241 }
242
243
244 public class Backgrounds{
245
246     public void go(){
247         final Timeline timeline=new Timeline(
248             new KeyFrame(
249                 Duration.seconds(1),
250                 event -> {
251                     if(staticData.tempSwitch){
252                         staticData.timerSeconds++;
253                     }
254                 }
255             )
256         );
257         timerLabel.setText(String.valueOf(staticData.timerSeconds));
258         // System.out.println(k);
259
260         int pt,second,minute;
261         second=staticData.timerSeconds;
262         pt=second/60;
263         minute=pt;
264         pt=60*pt;
265         second=second-pt;

```

```

266         timerLabel.setText(" "+minute+" :
"+second);
267
268
269         float tend,diff;
270         int fspeed;
271         tend=staticData.timerSeconds;
272         diff=tend/td.spaceCounter;
273         // difrnc=diff*time(tend,tstart);
274         //      System.out.println("tend: "+tend+"
diff :"+diff+" Space Counter :"+td.spaceCounter);
275         fspeed=(int) (60/diff);
276
277
speedLabel.setText(String.valueOf(fspeed));
278         if(staticData.timerSeconds%100==0){
279
if(td.user.usrdata.Score.get(staticData.usrname)<fspeed) {
280
td.user.usrdata.Score.replace(staticData.usrname,fspeed);
281         try {
282             td.user.writeUsers();
283         } catch (IOException e) {
284             e.printStackTrace();
285         }
286     }
287 }
288 }
289 }
290 )
291 );
292 timeline.setCycleCount(Animation.INDEFINITE);
293 timeline.play();
294 }
295
296 }
297
298
299
300     public void initializeHelper() throws IOException,
ClassNotFoundException {
301
td.user.readUsers();
302
if(!td.user.usrdata.Score.containsKey(staticData.usrname)) {
303
td.user.usrdata.Score.put(staticData.usrname,0);
304
}
305
306
307
308     Map<String, Integer> sortedMapAsc;
309     sortedMapAsc= sortByComparator(td.user.usrdata.Score, ASC);
310     String[] keysName = sortedMapAsc.keySet().toArray(new
String[0]);
311
312     highScoreLablename.setText(keysName[0]);
313
highScoreLable.setText(String.valueOf(td.user.usrdata.Score.get(keysName[0]))
);

```

```

314
315     Backgrounds backgrounds=new Backgrounds();
316     backgrounds.go();
317
318     td.t= new Thread(td.hw);
319
320     userNameLabel.setText(staticData.usrname);
321
322     String filename ;
323     filename=staticData.myDir+"/.src/lesson/"+td.lsnlist.get(0);
324     fileReaderClass f = new fileReaderClass(filename);
325     td.lsn= f.run();
326     td.toPrint="";
327     String[] lsnparts= td.lsn.split("");
328     for(int i = 0 ;i<20;i++){
329         td.toPrint=td.toPrint+lsnparts[i];
330     }
331     td.tempString="";
332     for(int i = 1 ;i<20;i++){
333         td.tempString=td.tempString+lsnparts[i];
334     }
335     task=td.toPrint;
336     // taskuser.setText(td.toPrint);
337     // userInput1.setText("_");
338     // userInput2.setText(td.tempString);
339     td.temp1=td.toPrint.split("");
340     taskC="_";
341     // System.out.println(taskC);
342     keys.clear();
343     keys.put(td.temp1[td.cursorPointer],1);
344     td.hw.write(keys,task,taskC);
345     td.t.run();
346
347
348 }
349
350
351 public void initialize() throws IOException, ClassNotFoundException
352 {
353     td.addLessons();
354
355
356
357
358     DevLevel.setText("By: Abdhesh Nayak");
359
360     tempKeyboard="file:///"+staticData.myDir+"/.src/Images/keyboard/tempKeyboard.h
361 tml";
362
363     td.user.readUsers();
364     if(!td.user.usrdata.Score.containsKey(staticData.usrname)){
365         td.user.usrdata.Score.put(staticData.usrname,0);
366     }
367
368     Map<String, Integer> sortedMapAsc;

```

```

368         sortedMapAsc= sortByComparator(td.user.usrdata.Score, ASC);
369         String[] keysName = sortedMapAsc.keySet().toArray(new
String[0]);
370
371         highScoreLablename.setText(keysName[0]);
372
highScoreLable.setText(String.valueOf(td.user.usrdata.Score.get(keysName[0]))
);
373
374         Backgrounds backgrounds=new Backgrounds();
375         backgrounds.go();
376
377         td.t= new Thread(td.hw);
378
379         userNameLabel.setText(staticData.usrname);
380
381         String filename ;
382         filename=staticData.myDir+"/.src/lesson/"+td.lsnlist.get(0);
383         fileReaderClass f = new fileReaderClass(filename);
384         td.lsn= f.run();
385         td.toPrint="";
386         String[] lsnparts= td.lsn.split("");
387         for(int i = 0 ;i<20;i++){
388             td.toPrint=td.toPrint+lsnparts[i];
389         }
390         td.tempString="";
391         for(int i = 1 ;i<20;i++){
392             td.tempString=td.tempString+lsnparts[i];
393         }
394         task=td.toPrint;
395         //         taskuser.setText(td.toPrint);
396         //         userInput1.setText("_");
397         //         userInput2.setText(td.tempString);
398         td.templ=td.toPrint.split("");
399         taskC="_";
400         //         System.out.println(taskC);
401         keys.clear();
402         keys.put(td.templ[td.cursorPointer],1);
403         td.hw.write(keys,task,taskC);
404         td.t.run();
405         setImageFun();
406
407         LanguageSelector.getItems().clear();
408         String[] lg = td.languages.values().toArray(new String[0]);
409         LanguageSelector.getItems().addAll(lg);
410
411         LanguageSelector.setValue("English");
412
413
414     }
415
416
417
418     public void onHandleKeyPressed(KeyEvent keyEvent) throws
IOException{
419         int x;
420         try {

```



```

421         x= (int) keyEvent.getText().charAt(0);
422     }catch (Exception e){
423         keys.clear();
424         keys.put(keyEvent.getCode().toString(),2);
425         keys.put(td.temp1[0],1);
426         td.hw.write(keys,task,taskC);
427         td.t.run();
428         keyboardLoad();
429         return;
430     }
431
432     if(x==8
433     ||x==127||x==16||(keyEvent.getText().equals("CAPS"))||(keyEvent.getText().equ
434     als("ESCAPE"))){
435         {
436             keys.clear();
437             keys.put(keyEvent.getCode().toString(),2);
438             keys.put(td.temp1[td.cursorPointer],1);
439             td.hw.write(keys,task,taskC);
440             td.t.run();
441             keyboardLoad();
442             return;
443         }
444         String[] lsnparts= td.lsn.split("");
445         String tempStr;
446         tempStr=keyEvent.getText();
447
448         if(keyEvent.isShiftDown()){
449             tempStr=td.is.convert(tempStr);
450         }
451         if(tempStr.equals(lsnparts[td.cursorPointer])){
452             if(!staticData.tempSwitch){
453                 staticData.tempSwitch=true;
454             }
455             if(lsnparts[td.cursorPointer].equals(" ")){
456                 td.spaceCounter++;
457             }
458             //td.spaceCounter++;
459             td.toPrint="";
460             if(td.cursorPointer==td.lsn.length()){
461                 td.lsnNext();
462             }
463             td.cursorPointer++;
464             if(td.cursorPointer%20==0) {
465                 td.lstcursorPointer=td.cursorPointer;
466                 for (int i = td.cursorPointer; i < td.cursorPointer +
467                 20; i++) {
468                     td.toPrint = td.toPrint + lsnparts[i];
469                 }
470                 task=td.toPrint;
471                 taskuser.setText(td.toPrint);
472             }
473
474             td.toPrint= "";
475             for (int i = td.lstcursorPointer; i < td.cursorPointer; i++)
476             {
477                 td.toPrint = td.toPrint + lsnparts[i];

```

```

474         }
475         td.toPrint=td.toPrint+"_";
476         taskC=td.toPrint;
477         //         userInput1.setText(td.toPrint);
478
479         td.toPrint= "";
480         for (int i = td.cursorPointer; i < td.lstcursorPointer+20;
i++) {
481             td.toPrint = td.toPrint + lsnparts[i];
482         }
483         td.tempString="";
484         for (int i = td.cursorPointer+1; i < td.lstcursorPointer+20;
i++) {
485             td.tempString=td.tempString + lsnparts[i];
486         }
487         //         userInput2.setText(td.tempString);
488         td.temp1=td.toPrint.split("");
489
490         if(td.temp1[0].equals(" ")){
491             td.temp1[0]="SPACE";
492         }
493
494         keys.clear();
495         keys.put(tempStr,0);
496         keys.put(td.temp1[0],1);
497
498         td.hw.write(keys,task,taskC);
499         td.t.run();
500
501         keyboardLoad();
502
503     }
504     else {
505         td.toPrint= "";
506         for (int i = td.lstcursorPointer; i < td.cursorPointer; i++)
{
507             td.toPrint = td.toPrint + lsnparts[i];
508         }
509         td.toPrint=td.toPrint+"_";
510         //         userInput1.setText(td.toPrint);
511
512         taskC=td.toPrint;
513         String temp;
514         temp= tempStr;
515
516         switch (keyEvent.getCode().toString()){
517             case "ESCAPE":
518                 temp="ESCAPE";
519                 break;
520             case "SPACE":
521                 temp="SPACE";
522                 break;
523             case "ENTER":
524                 temp="ENTER";
525                 break;
526             case "BACK_SPACE":
527                 temp="BACK_SPACE";

```

```

528             break;
529         case "TAB":
530             temp="TAB";
531
532     }
533
534     keys.clear();
535     keys.put(temp,2);
536     keys.put(td.temp1[0],1);
537     td.hw.write(keys,task,taskC);
538     td.t.run();
539     keyboardLoad();
540
541     td.toPrint= "";
542     for (int i = td.cursorPointer+1; i < td.lstcursorPointer+20;
543 i++) {
544         td.toPrint = td.toPrint + lsnparts[i];
545     }
546     // userinput2.setText(td.toPrint);
547 }
548
549
550     public void mouseClickedAction(MouseEvent mouseEvent) throws
551 IOException, ClassNotFoundException {
552         staticData.tempSwitch=false;
553         staticData.timerSeconds=0;
554         td.spaceCounter=0;
555
556         String filename ;
557         filename = staticData.myDir+"/.src/lesson/"+td.lsnlist.get(0);
558         if(mouseEvent.getSource().equals(level1)) {
559             filename =
560 staticData.myDir+"/.src/lesson/"+td.lsnlist.get(0);
561         }else if (mouseEvent.getSource().equals(level2)) {
562             filename =
563 staticData.myDir+"/.src/lesson/"+td.lsnlist.get(20);
564         }else if (mouseEvent.getSource().equals(level3)) {
565             filename =
566 staticData.myDir+"/.src/lesson/"+td.lsnlist.get(30);
567         }else if(mouseEvent.getSource().equals(btnNepali)){
568             td.cursorPointer=0;
569             WebEngine webEngine= webopen.getEngine();
570             staticData.keyBoardType=false;
571
572             staticData.unvLanguage=LanguageSelector.getValue().toString();
573             td.hw.ChangeLanguage(LanguageSelector.getValue().toString());
574             td.hw.run();
575             webEngine.load("file:///"+staticData.myDir+"/.src/Images/keyboard/tempKeyboard
.html");
576             //
577             //
578             return;
579         }else if (mouseEvent.getSource().equals(btnEnglish)) {

```

```

576 //          td.cursorPointer=0;
577          WebEngine webEngine = webopen.getEngine();
578          staticData.keyBoardType=true;
579 //          taskC="";
580 //          td.hw.write(keys,task,taskC);
581
582
583 staticData.unvLanguage=LanguageSelector.getValue().toString();
584
585 td.hw.ChangeLanguage(LanguageSelector.getValue().toString());
586 //          td.hw.run();
587 //
588 webEngine.load("file://" + staticData.myDir + "/.src/Images/keyboard/tempKeyboard
.html");
589 //          return;
590
591 }else
592 if(mouseEvent.getSource().equals(btnTop) || mouseEvent.getSource().equals(btnBo
ttom)) {
593         if(mouseEvent.getSource().equals(btnTop)) {
594             staticData.order=1;
595         }else{
596             staticData.order=2;
597         }
598         System.out.println("btnTop Clicked");
599         if(td.dialogptr) {
600             td.dialogptr = false;
601
602             Stage stage = new Stage();
603
604             FileInputStream input = new
605             FileInputStream(staticData.myDir + "/.src/Images/Icons/DialogIcon.png");
606             stage.getIcons().add(new Image(input));
607
608             BorderPane root;
609
610             root = (BorderPane)
611             FXMLLoader.load(getClass().getResource("topUsersView.fxml"));
612             Scene scene = new Scene(root, 800, 600);
613             stage.setScene(scene);
614             stage.showAndWait();
615             td.dialogptr=true;
616         }
617
618 }else if (mouseEvent.getSource().equals(btnAddUser)) {
619     userDialog("addUserWindow.fxml");
620     return;
621 }else if (mouseEvent.getSource().equals(btnStories)) {
622     if(td.dialogptr) {
623         staticData.keyBoardType=true;
624         td.dialogptr = false;
625         System.out.println("Free Button");
626
627         FileInputStream input = new
628         FileInputStream(staticData.myDir + "/.src/Images/Icons/DialogIcon.png");
629         stage.getIcons().add(new Image(input));

```

```

623 //          Stage stage = (Stage)
btnBottom.getScene().getWindow();
624          GridPane root;
625
626
627          root = (GridPane)
FXMLLoader.load(getClass().getResource("StoriesTyping.fxml"));
628          Scene scene = new Scene(root, 1280, 770);
629          stage.setScene(scene);
630
631
632
633          stage.showAndWait();
634
635
636
637          td.dialogptr=true;
638          staticData.timerSeconds=0;
639 //          StoriesTyping.backgrounds.timeline.stop();
640
641
642      }
643      }else if(mouseEvent.getSource().equals(btnFree)){
644          if(td.dialogptr) {
645              td.dialogptr = false;
646 //              System.out.println("Free Button");
647              Stage stage = new Stage();
648
649              FileInputStream input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/DialogIcon.png");
650              stage.getIcons().add(new Image(input));
651
652 //              Stage stage = (Stage)
btnBottom.getScene().getWindow();
653              GridPane root;
654
655
656              root = (GridPane)
FXMLLoader.load(getClass().getResource("FreeWriting.fxml"));
657              Scene scene = new Scene(root, 1280, 750);
658              stage.setScene(scene);
659
660
661              stage.showAndWait();
662
663              td.dialogptr=true;
664              staticData.timerSeconds=0;
665 //              StoriesTyping.backgrounds.timeline.stop();;
666
667          }
668      }else {
669          filename =
staticData.myDir+"/.src/lesson/"+td.lsnlist.get(0);
670      }
671      fileReaderClass f = new fileReaderClass(filename);
672      td.lsn= f.run();
673

```

```

674         td.toPrint="";
675         String[] lsnparts= td.lsn.split("");
676         for(int i = 0 ;i<20;i++){
677             td.toPrint=td.toPrint+lsnparts[i];
678         }
679         td.tempString="";
680         for(int i = 1 ;i<20;i++){
681             td.tempString=td.tempString+lsnparts[i];
682         }
683         task=td.toPrint;
684         // taskuser.setText(td.toPrint);
685         taskC="_";
686         // userInput2.setText(td.tempString);
687         // userInput1.setText("_");
688         // userInput3.setText("");
689         td.cursorPointer=0;
690         td.lstcursorPointer=0;
691         td.temp1=td.toPrint.split("");
692         keys.clear();
693         keys.put(td.temp1[td.cursorPointer],1);
694         td.hw.write(keys,task,taskC);
695         td.t.run();
696         keyboardLoad();
697     }
698
699
700     public void userDialog(String fxmlName) throws IOException,
ClassNotFoundException {
701         Dialog<ButtonType> dialog = new Dialog<>();
702         dialog.initOwner(MainGridPane.getScene().getWindow());
703         FXMLLoader fxmlLoader= new FXMLLoader();
704         fxmlLoader.setLocation(getClass().getResource(fxmlName));
705         try {
706             dialog.getDialogPane().setContent(fxmlLoader.load());
707         }catch (IOException e){
708             // System.out.println("Exception:");
709             e.printStackTrace();
710             return;
711         }
712
713         // Parent root =
FXMLLoader.load(getClass().getResource("addUserWindow.fxml"));
714         // dialog.getDialogPane().setContent(root);
715         // dialog.setTitle("Add New User");
716         dialog.getDialogPane().getButtonTypes().add(ButtonType.OK);
717         dialog.getDialogPane().getButtonTypes().add(ButtonType.CLOSE);
718         // System.out.println("Ok");
719         Optional<ButtonType> result = dialog.showAndWait();
720         if(result.isPresent() && result.get() == ButtonType.OK) {
721             addUserClass newUser = fxmlLoader.getController();
722             if(newUser.save() == 1) {
723                 result=dialog.showAndWait();
724             }
725
726         // System.out.println("OK Pressed");
727
728         }else {

```

```

729 //          System.out.println("Cancel Pressed");
730     }
731
732
733     }
734
735 }
736
737
738

```

## Mainwindow.fxml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.scene.control.Button?>
5  <?import javafx.scene.control.ChoiceBox?>
6  <?import javafx.scene.control.Label?>
7  <?import javafx.scene.control.Menu?>
8  <?import javafx.scene.control.MenuBar?>
9  <?import javafx.scene.control.MenuItem?>
10 <?import javafx.scene.image.ImageView?>
11 <?import javafx.scene.layout.BorderPane?>
12 <?import javafx.scene.layout.ColumnConstraints?>
13 <?import javafx.scene.layout.GridPane?>
14 <?import javafx.scene.layout.HBox?>
15 <?import javafx.scene.layout.RowConstraints?>
16 <?import javafx.scene.layout.VBox?>
17 <?import javafx.scene.web.WebView?>
18
19 <GridPane fx:id="MainGridPane" alignment="center" hgap="10" style="-fx-
background-color:white;" vgap="10" xmlns="http://javafx.com/javafx/8.0.211"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdheshnayak.typingguru.MainWindowController">
20     <VBox prefHeight="776.0" prefWidth="1410.0" style="-fx-font-size:
20;-fx-background-color: white;" GridPane.hgrow="ALWAYS"
GridPane.vgrow="ALWAYS">
21         <children>
22             <MenuBar VBox.vgrow="NEVER">
23                 <menus>
24                     <Menu mnemonicParsing="false" text="File">
25                         <items>
26                             <MenuItem fx:id="quite"
mnemonicParsing="false" onAction="#QuiteAction" text="Quit" />
27                         </items>
28                     </Menu>
29                     <Menu mnemonicParsing="false" text="About">
30                         <items>
31                             <MenuItem mnemonicParsing="false"
fx:id="aboutAppbtn" onAction="#AboutAppFun" text="About TypingGuru" />
32                         </items>
33                     </Menu>
34                 </menus>
35             </MenuBar>

```

```

36         <BorderPane prefHeight="499.0" prefWidth="902.0"
VBox.vgrow="ALWAYS">
37             <top>
38                 <HBox prefHeight="78.0"
BorderPane.alignment="CENTER">
39                     <children>
40                         <Button fx:id="btnHome" contentDisplay="TOP"
mnemonicParsing="false" onKeyPressed="#onHandleKeyPressed"
onMouseClicked="#mouseClickedAction" prefHeight="100.0" text="Home"
HBox.hgrow="ALWAYS">
41                             <graphic>
42                                 <ImageView fx:id="HomeIcon"
fitHeight="45.0" fitWidth="45.0" pickOnBounds="true" preserveRatio="true">
43                                     </ImageView>
44                                 </graphic>
45                             </Button>
46                         <Button fx:id="btnAddUser" contentDisplay="TOP"
mnemonicParsing="false" onKeyPressed="#onHandleKeyPressed"
onMouseClicked="#mouseClickedAction" prefHeight="100.0" text="Add User"
HBox.hgrow="ALWAYS">
47                             <graphic>
48                                 <ImageView fx:id="adduserIcon"
fitHeight="45.0" fitWidth="45.0" pickOnBounds="true" preserveRatio="true" />
49                                     </graphic>
50                             </Button>
51                         <Button fx:id="btnTop" contentDisplay="TOP"
mnemonicParsing="false" onKeyPressed="#onHandleKeyPressed"
onMouseClicked="#mouseClickedAction" prefHeight="100.0" text="Top"
HBox.hgrow="ALWAYS">
52                             <graphic>
53                                 <ImageView fx:id="topicon"
fitHeight="45.0" fitWidth="45.0" pickOnBounds="true" preserveRatio="true">
54                                     </ImageView>
55                                 </graphic>
56                             </Button>
57                         <Button fx:id="btnBottom"
contentDisplay="TOP" mnemonicParsing="false"
onKeyPressed="#onHandleKeyPressed" onMouseClicked="#mouseClickedAction"
prefHeight="100.0" text="Bottom" HBox.hgrow="ALWAYS">
58                             <graphic>
59                                 <ImageView fx:id="bottomIcon"
fitHeight="45.0" fitWidth="45.0" pickOnBounds="true" preserveRatio="true">
60                                     </ImageView>
61                                 </graphic>
62                             </Button>
63                         <Button fx:id="btnStories"
contentDisplay="TOP" mnemonicParsing="false"
onKeyPressed="#onHandleKeyPressed" onMouseClicked="#mouseClickedAction"
prefHeight="100.0" text="Stories" HBox.hgrow="ALWAYS">
64                             <graphic>
65                                 <ImageView fx:id="StoriesIcon"
fitHeight="45.0" fitWidth="45.0" pickOnBounds="true" preserveRatio="true">
66                                     </ImageView>
67                                 </graphic>
68                             </Button>
69                         <Button fx:id="btnFree" contentDisplay="TOP"
mnemonicParsing="false" onKeyPressed="#onHandleKeyPressed"

```



```

onMouseClicked="#mouseClickedAction" prefHeight="100.0" text="Free"
HBox.hgrow="ALWAYS">
70             <graphic>
71                 <ImageView fx:id="FreeIcon"
fitHeight="45.0" fitWidth="45.0" pickOnBounds="true" preserveRatio="true">
72                     </ImageView>
73             </graphic>
74         </Button>
75         <VBox prefHeight="79.0">
76             <children>
77                 <Button fx:id="level1"
minHeight="33.0" mnemonicParsing="false" onKeyPressed="#onHandleKeyPressed"
onMouseClicked="#mouseClickedAction" prefHeight="30.0" style="-fx-font-size:
15;" text="Level 1">
78                     <graphic>
79                         <ImageView
fx:id="level1Icon" fitHeight="15.0" fitWidth="15.0" pickOnBounds="true"
preserveRatio="true">
80                             </ImageView>
81                         </graphic>
82                     </Button>
83                     <Button fx:id="level2"
minHeight="33.0" mnemonicParsing="false" onKeyPressed="#onHandleKeyPressed"
onMouseClicked="#mouseClickedAction" prefHeight="30.0" style="-fx-font-size:
15;" text="Level 2">
84                         <graphic>
85                             <ImageView
fx:id="level2Icon" fitHeight="15.0" fitWidth="15.0" pickOnBounds="true"
preserveRatio="true">
86                                 </ImageView>
87                             </graphic>
88                         </Button>
89                         <Button fx:id="level3"
minHeight="33.0" mnemonicParsing="false" onKeyPressed="#onHandleKeyPressed"
onMouseClicked="#mouseClickedAction" style="-fx-font-size: 15;" text="Level
3">
90                             <graphic>
91                                 <ImageView
fx:id="level3Icon" fitHeight="15.0" fitWidth="15.0" pickOnBounds="true"
preserveRatio="true">
92                                     </ImageView>
93                                 </graphic>
94                             </Button>
95                         </children>
96                     </VBox>
97                 <VBox minWidth="-Infinity" prefHeight="99.0"
prefWidth="560.0" style="-fx-background-color: lavender;"
HBox.hgrow="ALWAYS">
98                     <children>
99                         <HBox prefHeight="100.0" prefWidth="200.0"
VBox.vgrow="ALWAYS">
100                             <children>
101                                 <ChoiceBox fx:id="LanguageSelector"
onAction="#LanguageController" prefWidth="200.0" style="-fx-border-color:
green;" />
102                                 <BorderPane HBox.hgrow="ALWAYS">
103                                     <center>

```

```

104                                     <Label fx:id="DevLevel" style="-
fx-font-size: 17; -fx-background-color: white;" text="By: Abdhesh Nayak"
textFill="#42269e" BorderPane.alignment="CENTER" />
105                                     </center>
106                                     </bottom>
107                                     <Label style="-fx-background-
color: white;-fx-font-size: 16;" text="Last Updated : 2019/08/10"
BorderPane.alignment="CENTER" />
108                                     </bottom>
109                                 </BorderPane>
110                            </children>
111                    </HBox>
112                    <VBox>
113                        <children>
114                            <HBox prefHeight="34.0"
prefWidth="699.0" VBox.vgrow="ALWAYS">
115                                <children>
116                                    <ImageView fx:id="userIcon"
fitHeight="35.0" fitWidth="40.0" pickOnBounds="true" preserveRatio="true" />
117                                    <VBox HBox.hgrow="ALWAYS">
118                                        <children>
119                                            <HBox>
120                                                <children>
121                                                    <Label style="-fx-
font-size: 17;" text=" Name : " />
122                                                        <Label
fx:id="userNameLabel" style="-fx-font-size: 17;" text="Alex Pardan" />
123                                                            </children>
124                                                        </HBox>
125                                                        <HBox>
126                                                            <children>
127                                                                <BorderPane
HBox.hgrow="ALWAYS">
128                                                                    <left>
129                                                                        <HBox
BorderPane.alignment="CENTER">
130                                                                            <children>
131                                                                                <Label
style="-fx-font-size: 18;" text=" Avg Speed : " />
132                                                                                    <Label
fx:id="speedLabel" style="-fx-font-size: 18;" text="33" />
133                                                                                        </children>
134                                                                                    </HBox>
135                                                                    </left>
136                                                                    <right>
137                                                                        <HBox
nodeOrientation="LEFT_TO_RIGHT" BorderPane.alignment="TOP_LEFT">
138                                                                            <children>
139                                                                                <Label
style="-fx-font-size: 18;" text="Hi Sc: " />
140                                                                                    <Label
fx:id="highScoreLablename" style="-fx-font-size: 18;" text="Abdhesh">
141
<HBox.margin>
142
<Insets left="5.0" right="5.0" />

```

```

143
144                                     </Label>
145                                     <Label
fx:id="highScoreLable" style="-fx-font-size: 18;" text="33">
146
147                                     </Label>
148                                     </children>
149                                     </HBox>
150                                     </right>
151                                     <center>
152                                     <HBox
alignment="CENTER" BorderPane.alignment="CENTER">
153                                     <children>
154                                     <Label
text="Timer:" textFill="#f50000" />
155                                     <Label
fx:id="timerLabel" text="0" />
156                                     </children>
157                                     </HBox>
158                                     </center>
159                                     </BorderPane>
160                                     </children>
161                                     </HBox>
162                                     </children>
163                                     </VBox>
164                                     </children>
165                                     </HBox>
166                                     </children>
167                                     </VBox>
168                                     </children>
169                                     </VBox>
170                                     </children>
171                                     </VBox>
172                                     </children>
173                                     </HBox>
174                                     </top>
175                                     <bottom>
176                                     <WebView fx:id="webopen" maxWidth="960.0"
minHeight="550.0" prefHeight="399.0" prefWidth="600.0"
BorderPane.alignment="BOTTOM_CENTER" />
177                                     </bottom>
178                                     </BorderPane>
179                                     </children>
180                                     </VBox>
181                                     <columnConstraints>
182                                     <ColumnConstraints />
183                                     </columnConstraints>
184                                     <rowConstraints>
185                                     <RowConstraints />
186                                     </rowConstraints>
187 </GridPane>
188

```

## passwordChecker.java

```
1  package com.abdheshnayak.typingguru;
2
3  import com.abdheshnayak.typingguru.JavaClasses.staticData;
4  import javafx.event.ActionEvent;
5  import javafx.fxml.FXML;
6  import javafx.fxml.FXMLLoader;
7  import javafx.scene.Scene;
8  import javafx.scene.control.Button;
9  import javafx.scene.control.Label;
10 import javafx.scene.control.PasswordField;
11 import javafx.scene.input.MouseEvent;
12 import javafx.scene.layout.GridPane;
13 import javafx.stage.Stage;
14
15 import java.io.IOException;
16
17 public class passwordChecker {
18     @FXML
19     private Label wrongLabel;
20     @FXML
21     private PasswordField passwordField;
22     @FXML
23     private Button closebtn;
24     @FXML
25     private Button loginbtn;
26     @FXML
27     private Label UserName;
28
29     public void initialize() {
30         UserName.setText(staticData.usrname);
31     }
32
33     public void onMouseclickHandle(MouseEvent mouseEvent) throws
IOException, ClassNotFoundException {
34         if (mouseEvent.getSource().equals(closebtn)) {
35             Stage stage = (Stage) closebtn.getScene().getWindow();
36             stage.close();
37
38             //         Stage stage = new Stage();
39             GridPane root;
40             root = (GridPane)
FXMLLoader.load(getClass().getResource("UserSelector.fxml"));
41             //         PasswordChecker ps = new PasswordChecker();
42             //         ps.usr= choicebox.getValue().toString();
43             Scene scene = new Scene(root, 600, 400);
44             stage.setScene(scene);
45             stage.show();
46         } else if (mouseEvent.getSource().equals(loginbtn)) {
47
48             addUserClass usr = new addUserClass();
49             usr.readUsers();
50             if (usr.usrdata.users.containsKey(staticData.usrname)) {
```

```

51         if
(passwordField.getText().trim().equals(usr.usrdata.users.get(staticData.usrname))) {
52             Stage stage = (Stage)
loginbtn.getScene().getWindow();
53             stage.close();
54             //             Stage stage = new Stage();
55             GridPane root;
56             root = (GridPane)
FXMLLoader.load(getClass().getResource("Mainwindow.fxml"));
57             //             PasswordChecker ps = new
PasswordChecker();
58             //             ps.usr=
choicebox.getValue().toString();
59             Scene scene = new Scene(root, 1280, 720);
60             stage.setScene(scene);
61             stage.show();
62         }else {
63
64             wrongLabel.setText("Wrong Password Try Again !!");
65             passwordField.clear();
66         }
67     }
68 }
69 }
70
71     public void onEnterClicked(ActionEvent event) throws IOException,
ClassNotFoundException {
72         addUserClass usr = new addUserClass();
73         usr.readUsers();
74         if (usr.usrdata.users.containsKey(staticData.username)) {
75
76             if
(passwordField.getText().trim().equals(usr.usrdata.users.get(staticData.usrname))) {
77
78                 Stage stage = (Stage) loginbtn.getScene().getWindow();
79                 stage.close();
80                 //             Stage stage = new Stage();
81                 GridPane root;
82                 root = (GridPane)
FXMLLoader.load(getClass().getResource("Mainwindow.fxml"));
83                 //             PasswordChecker ps = new
PasswordChecker();
84                 //             ps.usr= choicebox.getValue().toString();
85                 Scene scene = new Scene(root, 1280, 720);
86                 stage.setScene(scene);
87                 stage.show();
88
89             }else {
90                 wrongLabel.setText("Wrong Password Try Again !!");
91                 passwordField.clear();
92             }
93         }
94     }
95 }

```

## passwordChecker.fxml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.scene.control.Button?>
5  <?import javafx.scene.control.Label?>
6  <?import javafx.scene.control.PasswordField?>
7  <?import javafx.scene.layout.BorderPane?>
8  <?import javafx.scene.layout.ColumnConstraints?>
9  <?import javafx.scene.layout.GridPane?>
10 <?import javafx.scene.layout.HBox?>
11 <?import javafx.scene.layout.RowConstraints?>
12 <?import javafx.scene.layout.VBox?>
13 <?import javafx.scene.text.Font?>
14
15 <GridPane alignment="CENTER" prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/8.0.211" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdheshnayak.typingguru.passwordChecker">
16     <BorderPane maxHeight="200.0" maxWidth="500.0" prefHeight="200.0"
prefWidth="500.0">
17         <bottom>
18             <BorderPane prefWidth="200.0" BorderPane.alignment="CENTER">
19                 <right>
20                     <HBox alignment="BOTTOM_RIGHT"
BorderPane.alignment="CENTER">
21                         <children>
22                             <Button fx:id="closebtn" mnemonicParsing="false"
onMouseClicked="#onMouceclickHandle" text="Close">
23                                 <HBox.margin>
24                                     <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
25                                 </HBox.margin>
26                             </Button>
27
28                             <Button fx:id="loginbtn"
onMouseClicked="#onMouceclickHandle" text="Login">
29                                 <HBox.margin>
30                                     <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
31                                 </HBox.margin></Button>
32                         </children>
33                     </HBox>
34                 </right>
35             </BorderPane>
36         </bottom>
37         <top>
38             <Label fx:id="UserName" text="User Name" textFill="#121c83"
BorderPane.alignment="CENTER">
39                 <BorderPane.margin>
40                     <Insets top="20.0" />
41                 </BorderPane.margin>
42                 <font>
43                     <Font name="System Bold Italic" size="24.0" />
44                 </font></Label>
45             </top>
46         </center>
```

```

47         <VBox alignment="CENTER" prefHeight="200.0" prefWidth="100.0"
BorderPane.alignment="CENTER">
48             <children>
49                 <Label fx:id="wrongLabel" textFill="RED">
50                     <VBox.margin>
51                         <Insets bottom="10.0" left="10.0" right="10.0"
top="10.0" />
52                     </VBox.margin>
53                     <font>
54                         <Font name="System Bold" size="18.0" />
55                     </font>
56                 </Label>
57                 <HBox alignment="CENTER" prefHeight="100.0"
prefWidth="200.0">
58                     <children>
59                         <Label text="Password : " />
60                         <PasswordField fx:id="passwordField"
onAction="#onEnterClicked" />
61                     </children>
62                 </HBox>
63             </children>
64         </VBox>
65     </center>
66 </BorderPane>
67 <columnConstraints>
68     <ColumnConstraints />
69 </columnConstraints>
70 <rowConstraints>
71     <RowConstraints />
72 </rowConstraints>
73 </GridPane>
74

```

#### addUserClass.java

```

1  package com.abdheshnayak.typingguru;
2
3
4  import com.abdheshnayak.typingguru.JavaClasses.*;
5  import javafx.fxml.FXML;
6  import javafx.scene.control.Label;
7  import javafx.scene.control.PasswordField;
8  import javafx.scene.control.TextField;
9
10 import java.io.*;
11 import java.util.HashMap;
12 import java.util.Map;
13 import java.util.Scanner;
14
15 public class addUserClass {
16     @FXML
17     private Label headLabel;
18     @FXML
19     private PasswordField password;

```

```

20     @FXML
21     private PasswordField password1;
22     @FXML
23     private TextField nameField;
24
25     public usrData usrdata = new usrData();
26     public langClass langclass = new langClass();
27     public void initialize(){
28         //      System.out.println("Hello");
29     }
30
31     private String pname=new
String(staticData.myDir+"/.src/UserData/userData.data");
32     private String langListFileName=new
String(staticData.myDir+"/.src/UserData/LanguageList.list");
33
34     public void readUsers() throws IOException, ClassNotFoundException {
35
36         String lsn= new String();
37         FileInputStream is = new FileInputStream(pname);
38
39         ObjectInputStream ois= new ObjectInputStream(is);
40         Map<String ,String> obj = new HashMap<>();
41
42         try {
43             usrdata=(usrData) ois.readObject();
44         }catch (Exception e){
45             System.out.println(e);
46         }finally {
47             try {
48                 is.close();
49                 ois.close();
50             }catch (Exception a){
51                 System.out.println(a);
52             }
53         }
54         //      System.out.println(usrdata.users.entrySet().toString());
55         //      while ((c = inputStream.read()) != -1) {
56         //          lsn=lsn+String.valueOf((char)c);
57         //      }
58         //      System.out.println(c);
59         //      x++;
60         //      }
61         //      } catch (FileNotFoundException e) {
62         //          e.printStackTrace();
63         //      } catch (IOException e) {
64         //          e.printStackTrace();
65         //      } finally {
66         //          if (inputStream != null) {
67         //              inputStream.close();
68         //          }
69         //      }
70         //      System.out.println(x);
71         //      return lsn;
72     }
73     public void readLanguages() throws IOException {
74         FileInputStream is = new FileInputStream(langListFileName);

```



```

75
76     ObjectInputStream ois= new ObjectInputStream(is);
77     Map<String ,String> obj = new HashMap<>();
78
79     try {
80         langclass=(langClass) ois.readObject();
81     }catch (Exception e){
82         System.out.println(e);
83     }finally {
84         try {
85             is.close();
86             ois.close();
87         }catch (Exception a){
88             System.out.println(a);
89         }
90     }
91 }
92 public void writeUsers() throws IOException {
93
94
95     FileOutputStream outputStream = new FileOutputStream(pname);
96     ObjectOutputStream oos = new ObjectOutputStream(outputStream);
97     try {
98         oos.writeObject(usrddata);
99     }catch (Exception e) {
100         e.printStackTrace();
101     } finally {
102         if (outputStream != null) {
103             try {
104                 outputStream.close();
105             } catch (IOException e) {
106                 e.printStackTrace();
107             }
108         }
109         if(oos != null){
110             try {
111                 oos.close();
112             }catch (IOException e){
113                 e.printStackTrace();
114             }
115         }
116     }
117     // System.out.println("Written User");
118 }
119
120
121 public void WriteLanguage() throws IOException {
122
123
124     int n;
125     String s;
126     Scanner scanner = new Scanner(System.in);
127     System.out.println("Enter the number of Language");
128     n= scanner.nextInt();
129     scanner.nextLine();
130     for (int i=0;i<n;i++){
131         System.out.println("Enter:");

```

```

132
133 langclass.languages.put(langclass.languages.size(), scanner.nextLine());
134     }
135     FileOutputStream outputStream = new
136     FileOutputStream(langListFileName);
137     ObjectOutputStream oos = new ObjectOutputStream(outputStream);
138     try {
139         oos.writeObject(langclass);
140     } catch (Exception e) {
141         e.printStackTrace();
142     } finally {
143         if (outputStream != null) {
144             try {
145                 outputStream.close();
146             } catch (IOException e) {
147                 e.printStackTrace();
148             }
149         }
150         if(oos != null){
151             try {
152                 oos.close();
153             } catch (IOException e){
154                 e.printStackTrace();
155             }
156         }
157     } // System.out.println("Written User");
158 }
159
160
161
162 public int save() throws IOException, ClassNotFoundException {
163     readUsers();
164     // System.out.println("Readed Data:
165     "+usrdata.users.entrySet().toString());
166     if(usrdata.users.containsKey(nameField.getText().trim())){
167         headLabel.setText("User Already Present \nTry with Another
168         Name");
169         return 1;
170     }
171     else
172     if(password.getText().trim().equals(password1.getText().trim())){
173         if(!usrdata.users.containsKey(nameField.getText().trim())) {
174             usrdata.users.put(nameField.getText().trim(),
175             password.getText().trim());
176             usrdata.Score.put(nameField.getText().trim(), 0);
177             usrdata.userNames.add(nameField.getText().trim());
178             writeUsers();
179         }
180     } // System.out.println("Tried To Write:
181     "+usrdata.users.entrySet().toString());
182     return 0;
183 } else {
184     // System.out.println("Password Not Matched");
185     headLabel.setText("Password Not Matched");

```

```

182         return 1;
183     //    }
184 }
185
186
187 }
188 }
189

```

### addUserWindow.fxml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.scene.control.DialogPane?>
4  <?import javafx.scene.control.Label?>
5  <?import javafx.scene.control.PasswordField?>
6  <?import javafx.scene.control.TextField?>
7  <?import javafx.scene.layout.ColumnConstraints?>
8  <?import javafx.scene.layout.GridPane?>
9  <?import javafx.scene.layout.RowConstraints?>
10 <?import javafx.scene.paint.RadialGradient?>
11 <?import javafx.scene.paint.Stop?>
12
13 <DialogPane fx:id="addNewUserDialog" minHeight="300.0" minWidth="500.0"
style="-fx-background-color: pink;" xmlns="http://javafx.com/javafx/8.0.211"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdreshnayak.typingguru.addUserClass">
14     <header>
15         <Label fx:id="headLabel" alignment="CENTER"
contentDisplay="CENTER" style="-fx-font-size: 25;" text="Add User">
16             <textFill>
17                 <RadialGradient centerX="0.6292134831460674"
centerY="0.46634615384615385" focusAngle="-150.95"
radius="0.47619047619047616">
18                     <stops>
19                         <Stop color="BLACK" />
20                         <Stop color="#0304ff" offset="1.0" />
21                     </stops>
22                 </RadialGradient>
23             </textFill></Label>
24     </header>
25     <content>
26         <GridPane>
27             <columnConstraints>
28                 <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0"
prefWidth="100.0" />
29                 <ColumnConstraints hgrow="SOMETIMES" minWidth="10.0"
prefWidth="100.0" />
30             </columnConstraints>
31             <rowConstraints>
32                 <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
33                 <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />

```

```

34         <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
35         <RowConstraints minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
36     </rowConstraints>
37     <children>
38         <TextField fx:id="nameField" promptText="Your Name" style="-
fx-font-size: 18;" GridPane.columnIndex="1" GridPane.hgrow="ALWAYS"
GridPane.vgrow="ALWAYS" />
39         <PasswordField fx:id="password" promptText="Password"
style="-fx-font-size: 18;" GridPane.columnIndex="1" GridPane.rowIndex="1"
GridPane.vgrow="ALWAYS" />
40         <Label style="-fx-font-size: 18;" text="Name :" />
41         <Label style="-fx-font-size: 18;" text="Password :"
GridPane.rowIndex="1" />
42         <Label style="-fx-font-size: 18;" text="Confirm Password"
GridPane.rowIndex="2" />
43         <PasswordField fx:id="password1" promptText="Confirm
Password" style="-fx-font-size: 18;" GridPane.columnIndex="1"
GridPane.rowIndex="2" />
44     </children>
45 </GridPane>
46 </content>
47 </DialogPane>
48

```

### DeleteUser.java

```

1  package com.abdheshnayak.typingguru;
2
3  import com.abdheshnayak.typingguru.JavaClasses.staticData;
4  import javafx.event.ActionEvent;
5  import javafx.fxml.FXML;
6  import javafx.scene.control.Button;
7  import javafx.scene.control.Label;
8  import javafx.scene.control.PasswordField;
9  import javafx.stage.Stage;
10
11 import java.io.IOException;
12
13 public class DeleteUser {
14     @FXML
15     private Label wrongpasswordlabel;
16     @FXML
17     private PasswordField passwordField;
18     @FXML
19     private Button btnYes;
20     @FXML
21     private Button btnNo;
22     @FXML
23     private Label usernameLabel;
24     addUserClass user= new addUserClass();
25     public void initialize(){
26

```

```

27         usernameLabel.setText(staticData.usrname);
28
29     }
30
31     public void onActionHandle(ActionEvent event) throws IOException,
ClassNotFoundException {
32         //
33         System.out.println(user.usrdata.users.get(user.usrdata.users.keySet().toArray
34         ())).);
35         if(event.getSource().equals(btnNo)){
36             Stage stage = (Stage) btnNo.getScene().getWindow();
37             stage.close();
38         }else if(event.getSource().equals(btnYes)){
39             //
40             System.out.println("Yes");
41
42             user.readUsers();
43             if (user.usrdata.users.containsKey(staticData.usrname)) {
44                 //
45                 String[] keysName =
46                 user.usrdata.users.keySet().toArray(new String[0]);
47                 //
48                 System.out.println(user.usrdata.userNames.get(0));
49                 if
50                 (passwordField.getText().trim().equals(user.usrdata.users.get(staticData.usrn
51                 ame)) || passwordField.getText().trim().equals(user.usrdata.users.get(user.usrd
52                 ata.userNames.get(0)))) {
53                     //
54                     System.out.println("Password Matched");
55                     user.usrdata.users.remove(staticData.usrname);
56                     user.usrdata.Score.remove(staticData.usrname);
57                     user.usrdata.userNames.remove(staticData.usrname);
58                     Stage stage = (Stage) btnNo.getScene().getWindow();
59                     stage.close();
60                     user.writeUsers();
61
62                 }else {
63                     wrongpasswordlabel.setText("Wrong Password Try
64                     Again");
65                     passwordField.clear();
66                     //
67                     System.out.println("Password Not Matched");
68                 }
69             }
70         }else {
71
72         }
73     }
74 }

```

### deleteUser.fxml

```

1    <?xml version="1.0" encoding="UTF-8"?>
2
3    <?import javafx.geometry.Insets?>
4    <?import javafx.scene.control.Button?>
5    <?import javafx.scene.control.Label?>
6    <?import javafx.scene.control.PasswordField?>

```

```

7      <?import javafx.scene.layout.BorderPane?>
8      <?import javafx.scene.layout.ColumnConstraints?>
9      <?import javafx.scene.layout.GridPane?>
10     <?import javafx.scene.layout.HBox?>
11     <?import javafx.scene.layout.RowConstraints?>
12     <?import javafx.scene.layout.VBox?>
13     <?import javafx.scene.text.Font?>
14
15     <GridPane alignment="TOP_CENTER"
xmlns="http://javafx.com/javafx/8.0.211" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdheshnayak.typingguru.DeleteUser">
16         <BorderPane minWidth="600.0">
17             <bottom>
18                 <HBox alignment="BOTTOM_RIGHT" BorderPane.alignment="CENTER">
19                     <children>
20                         <Button fx:id="btnNo" mnemonicParsing="false"
onAction="#onActionHandle" text="No" textFill="#126500">
21                             <HBox.margin>
22                                 <Insets bottom="10.0" left="10.0" right="10.0"
top="10.0" />
23                             </HBox.margin>
24                         </Button>
25                         <Button fx:id="btnYes" mnemonicParsing="false"
onAction="#onActionHandle" text="Yes" textFill="#c91b1b">
26                             <HBox.margin>
27                                 <Insets bottom="10.0" left="10.0" right="10.0"
top="10.0" />
28                             </HBox.margin>
29                         </Button>
30                     </children>
31                     <opaqueInsets>
32                         <Insets />
33                     </opaqueInsets>
34                 </HBox>
35             </bottom>
36             <center>
37                 <VBox BorderPane.alignment="CENTER">
38                     <children>
39                         <VBox alignment="CENTER">
40                             <children>
41                                 <HBox alignment="CENTER">
42                                     <children>
43                                         <Label text="Are You Sure want To delete ?"
textFill="#f21a1a">
44                                             <HBox.margin>
45                                                 <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
46                                             </HBox.margin>
47                                             <font>
48                                                 <Font size="18.0" />
49                                             </font>
50                                         </Label>
51                                     </children>
52                                 </HBox>
53                                 <Label fx:id="usernameLabel" alignment="CENTER"
contentDisplay="CENTER" text="User Name" textFill="#5b2703">
54                                     <font>

```

```

55         <Font size="24.0" />
56     </font>
57 </Label>
58 <HBox alignment="CENTER">
59     <children>
60         <Label fx:id="wrongpasswordlabel"
textFill="#c4529f">
61             <font>
62                 <Font size="19.0" />
63             </font>
64         </Label>
65     </children>
66     <VBox.margin>
67         <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
68     </VBox.margin>
69 </HBox>
70 <HBox alignment="CENTER">
71     <children>
72         <Label text="Your / Admin Password">
73             <font>
74                 <Font size="15.0" />
75             </font>
76         <HBox.margin>
77             <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
78         </HBox.margin>
79     </Label>
80     <PasswordField fx:id="passwordField"
promptText="Password">
81         <font>
82             <Font size="14.0" />
83         </font>
84         <HBox.margin>
85             <Insets bottom="10.0" right="10.0"
top="10.0" />
86         </HBox.margin>
87     </PasswordField>
88 </children>
89 <VBox.margin>
90     <Insets bottom="20.0" left="20.0"
right="20.0" top="20.0" />
91 </VBox.margin>
92 </HBox>
93 </children>
94 </VBox>
95 </children>
96 <BorderPane.margin>
97     <Insets bottom="20.0" left="20.0" right="20.0" top="30.0"
/>
98 </BorderPane.margin>
99 </VBox>
100 </center>
101 </BorderPane>
102 <columnConstraints>
103     <ColumnConstraints />
104 </columnConstraints>

```

```
105     <rowConstraints>
106         <RowConstraints />
107     </rowConstraints>
108 </GridPane>
109
```

## FreeWriting.java

```
1  package com.abdheshnayak.typingguru;
2
3  import com.abdheshnayak.typingguru.JavaClasses.fileReaderClass;
4  import com.abdheshnayak.typingguru.JavaClasses.staticData;
5  import com.abdheshnayak.typingguru.JavaClasses.tempDatas;
6  import javafx.event.ActionEvent;
7  import javafx.fxml.FXML;
8  import javafx.scene.control.Button;
9  import javafx.scene.control.ChoiceBox;
10 import javafx.scene.control.Label;
11 import javafx.scene.image.Image;
12 import javafx.scene.image.ImageView;
13 import javafx.scene.input.KeyEvent;
14 import javafx.scene.input.MouseEvent;
15 import javafx.scene.web.WebEngine;
16 import javafx.scene.web.WebView;
17
18 import java.io.FileInputStream;
19 import java.io.IOException;
20 import java.util.HashMap;
21 import java.util.Map;
22 import java.util.Random;
23
24
25 public class FreeWriting {
26     @FXML
27     private ImageView playpauseImage;
28     @FXML
29     private Button powerBtn;
30     @FXML
31     private ChoiceBox LanguageSelector;
32     @FXML
33     private WebView toTypeViewLayoutWebView;
34     @FXML
35     private Label userNameLabel;
36     @FXML
37     private WebView webopen;
38     private Map<String, Integer> keys = new HashMap<>();
39
40     private boolean pwrBtnFlag=false;
41
42     private tempDatas td = new tempDatas();
43
44     private String taskC;
45     private String strWord = "";
46
```



```

47
48     private String tempKeyboardtoType="file://" +
staticData.myDir+"/.src/Images/keyboard/tempToTypehtmlfile.html";
49     private String
tempKeyboard="file://" +staticData.myDir+"/.src/Images/keyboard/tempKeyboard.h
tml";
50
51
52
53     // public void LanguageController(ActionEvent actionEvent) throws
IOException, ClassNotFoundException {
54     //
55     //         if(actionEvent.getSource().equals(LanguageSelector)) {
56     //
staticData.unvLanguage=LanguageSelector.getValue().toString();
57     //
td.hw.ChangeLanguage(LanguageSelector.getValue().toString());
58     //         td.t.run();
59     //         keyboardLoad();
60     //     }
61     //
62     //     }
63
64
65     public void LanguageController(ActionEvent actionEvent) {
66         if(actionEvent.getSource().equals(LanguageSelector)) {
67             strWord="";
68
staticData.unvLanguage=LanguageSelector.getValue().toString();
69
td.hw.ChangeLanguage(LanguageSelector.getValue().toString());
70             td.t.run();
71             playpauseImage.setImage(image);
72             powerBtn.setText("Start");
73             pwrBtnFlag=false;
74
75             td.tw.write("Type Here....After Clicking On Start Button.");
76             td.tl.run();
77
78             keyboardLoad();
79         }
80
81     }
82
83     FileInputStream input;
84     Image image ,image2;
85     public void initialize() throws IOException {
86         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/play.png");
87         image = new Image(input);
88         input = new
FileInputStream(staticData.myDir+"/.src/Images/Icons/pause.png");
89         image2= new Image(input);
90         playpauseImage.setImage(image);
91
92
93         powerBtn.setText("Start");

```

```

94
95         td.addLessons();
96
97         td.t = new Thread(td.hw);
98
99         td.hw.setToNull();
100         td.hw.ChangeLanguage(staticData.unvLanguage);
101
102
103
104
105
106 //         td.user.readLanguages();
107         LanguageSelector.getItems().clear();
108         String[] lg = td.languages.values().toArray(new String[0]);
109         LanguageSelector.getItems().addAll(lg);
110
111 //         LanguageSelector.setValue("English");
112
113         userNameLabel.setText(staticData.usrname);
114
115
116         td.t1= new Thread(td.tw);
117
118
119         td.tw.write("Type Here");
120         td.t1.run();
121
122
123         keys.clear();
124         td.hw.write(keys);
125         td.t.run();
126
127         keyboardLoad();
128
129         LanguageSelector.setValue("English");
130
131     }
132
133     private void keyboardLoad() {
134         WebEngine webEngine= webopen.getEngine();
135         webEngine.load(tempKeyboard);
136
137         WebEngine webEngine1= toTypeViewLayoutWebView.getEngine();
138         webEngine1.load(tempKeyboardtoType);
139     }
140
141
142     public void onHandleKeyPressed(KeyEvent keyEvent) throws IOException
143     {
144         if(pwrBtnFlag==true) {
145             int x;
146
147             //         System.out.println("Size = " + strWord.length());
148             //         System.out.println(keyEvent.toString());
149             //         System.out.println(keyEvent.getCode());
150             switch (keyEvent.getCode().toString()) {

```

```

150         case "CAPS":
151             if (keys.containsKey("CAPS")) {
152                 keys.remove(keyEvent.getCode().toString());
153                 td.hw.write(keys);
154                 td.t.run();
155                 keyboardLoad();
156                 return;
157             }
158         case "ALT":
159         case "TAB":
160         case "SHIFT":
161         case "ENTER":
162         case "ESCAPE":
163         case "CONTROL":
164             //                System.out.println("hello");
165             keys.clear();
166             keys.put(keyEvent.getCode().toString(), 1);
167             td.hw.write(keys);
168             td.t.run();
169             keyboardLoad();
170             return;
171         case "BACK_SPACE":
172             keys.clear();
173             if (strWord.length() == 0) {
174                 keys.put(keyEvent.getCode().toString(), 2);
175             } else {
176                 keys.put(keyEvent.getCode().toString(), 1);
177
178                 strWord = strWord.substring(0, strWord.length()
179 - 1);
180
181                 td.tw.write(strWord);
182                 td.tl.run();
183             }
184             td.hw.write(keys);
185             td.t.run();
186             keyboardLoad();
187             return;
188         }
189         String tempStr;
190         tempStr = keyEvent.getText();
191
192         if (keyEvent.isShiftDown()) {
193             tempStr = td.is.convert(tempStr);
194         }
195
196         strWord = strWord + tempStr;
197
198         //                System.out.println(keys.size());
199         keys.put(tempStr, 1);
200         if (keys.size() > 1) {
201             keys.clear();
202             keys.put(tempStr, 1);
203         }
204         td.t.run();
205         td.tw.write(strWord);
206         td.tl.run();
207         keyboardLoad();

```

```

206     }
207 }
208
209 public void powerBtnController(MouseEvent mouseEvent) {
210     if(mouseEvent.getSource().equals(powerBtn)){
211         if(pwrBtnFlag==true){
212             pwrBtnFlag=false;
213             playpauseImage.setImage(image);
214             powerBtn.setText("Start");
215
216         }else {
217             pwrBtnFlag=true;
218             playpauseImage.setImage(image2);
219             powerBtn.setText("Pause");
220
221         }
222     }
223 }
224 }
225

```

### FreeWriting.fxml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.geometry.Rectangle2D?>
5  <?import javafx.scene.control.Button?>
6  <?import javafx.scene.control.ChoiceBox?>
7  <?import javafx.scene.control.Label?>
8  <?import javafx.scene.image.ImageView?>
9  <?import javafx.scene.layout.BorderPane?>
10 <?import javafx.scene.layout.ColumnConstraints?>
11 <?import javafx.scene.layout.GridPane?>
12 <?import javafx.scene.layout.HBox?>
13 <?import javafx.scene.layout.RowConstraints?>
14 <?import javafx.scene.layout.VBox?>
15 <?import javafx.scene.paint.RadialGradient?>
16 <?import javafx.scene.paint.Stop?>
17 <?import javafx.scene.text.Font?>
18 <?import javafx.scene.web.WebView?>
19
20 <GridPane fx:id="MainGridPane" alignment="center" hgap="10" style="-fx-
background-color:white;" vgap="10" xmlns="http://javafx.com/javafx/8.0.211"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdheshnayak.typingguru.FreeWriting">
21     <VBox prefHeight="776.0" prefWidth="1410.0">
22         <children>
23             <BorderPane>
24                 <center>
25                     <HBox alignment="CENTER" prefHeight="90.0"
prefWidth="1179.0" BorderPane.alignment="CENTER">
26                         <children>
27                             <Label alignment="CENTER"
contentDisplay="CENTER" text="Free Typing">

```

```

28             <textFill>
29                 <RadialGradient centerX="0.5"
centerY="0.5" radius="0.5">
30                     <stops>
31                         <Stop color="RED" />
32                         <Stop color="#610eae"
offset="1.0" />
33                     </stops>
34                 </RadialGradient>
35             </textFill>
36             <font>
37                 <Font size="43.0" />
38             </font>
39         </Label>
40     </children>
41 </HBox>
42 </center>
43 <left>
44     <HBox alignment="CENTER"
BorderPane.alignment="CENTER">
45         <children>
46             <Label fx:id="userNameLabel"
minWidth="200.0" text="Label" />
47         </children>
48         <BorderPane.margin>
49             <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
50         </BorderPane.margin>
51     </HBox>
52 </left>
53 <right>
54     <HBox alignment="CENTER" minWidth="150.0"
BorderPane.alignment="CENTER">
55         <BorderPane.margin>
56             <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
57         </BorderPane.margin>
58     <children>
59         <ChoiceBox fx:id="LanguageSelector"
onAction="#LanguageController" prefWidth="150.0">
60             <HBox.margin>
61                 <Insets right="5.0" />
62             </HBox.margin>
63         </ChoiceBox>
64         <Button fx:id="powerBtn" mnemonicParsing="false"
onKeyPressed="#onHandleKeyPressed" onMouseClicked="#powerBtnController">
65             <HBox.margin>
66                 <Insets left="10.0" />
67             </HBox.margin>
68             <graphic>
69                 <ImageView fx:id="playpauseImage"
fitHeight="20.0" fitWidth="20.0" pickOnBounds="true" preserveRatio="true"
scaleX="1.5" scaleY="1.5">
70                     <viewport>
71                         <Rectangle2D />
72                     </viewport>
73                 </ImageView>

```

```

74         </graphic>
75     </Button>
76 </children>
77 </HBox>
78 </right>
79 </BorderPane>
80 <BorderPane prefHeight="499.0" prefWidth="902.0"
VBox.vgrow="ALWAYS">
81     <bottom>
82         <WebView fx:id="webopen" maxWidth="960.0"
minHeight="460.0" onKeyPressed="#onHandleKeyPressed" prefHeight="399.0"
prefWidth="600.0" BorderPane.alignment="BOTTOM_CENTER" />
83     </bottom>
84     <top>
85         <WebView fx:id="toTypeViewLayoutWebView"
maxHeight="240.0" onKeyPressed="#onHandleKeyPressed" maxWidth="900.0"
minHeight="240.0" minWidth="800.0" prefHeight="-1.0" prefWidth="-1.0"
BorderPane.alignment="CENTER" />
86     </top>
87 </BorderPane>
88 </children>
89 </VBox>
90 <columnConstraints>
91     <ColumnConstraints />
92 </columnConstraints>
93 <rowConstraints>
94     <RowConstraints />
95 </rowConstraints>
96 </GridPane>
97

```

## StoriesTyping.java

```

1  package com.abdreshnayak.typingguru;
2
3  import com.abdreshnayak.typingguru.JavaClasses.fileReaderClass;
4  import com.abdreshnayak.typingguru.JavaClasses.staticData;
5  import com.abdreshnayak.typingguru.JavaClasses.tempDatas;
6  import javafx.animation.Animation;
7  import javafx.animation.KeyFrame;
8  import javafx.animation.Timeline;
9  import javafx.application.Application;
10 import javafx.event.Event;
11 import javafx.event.EventHandler;
12 import javafx.event.EventType;
13 import javafx.fxml.FXML;
14 import javafx.scene.control.Label;
15 import javafx.scene.input.KeyEvent;
16 import javafx.scene.input.MouseEvent;
17 import javafx.scene.web.WebEngine;
18 import javafx.scene.web.WebView;
19 import javafx.stage.Stage;
20 import javafx.stage.WindowEvent;
21 import javafx.util.Duration;

```

```

22
23 import java.io.IOException;
24 import java.util.HashMap;
25 import java.util.Map;
26 import java.util.Random;
27
28
29 public class StoriesTyping {
30     @FXML
31     private WebView toTypeViewLayoutWebView;
32     // @FXML
33     // private Label userTyping;
34
35     @FXML
36     private Label userNameLabel;
37
38     @FXML
39     private WebView webopen;
40     private int tempcursorpointer=0;
41     private int ptr,tempptr=0;
42     private Map<String, Integer> keys = new HashMap<>();
43
44     private int tempspaceCounter;
45     private String[] lsnparts;
46     private String[] lsnparts1;
47     private tempDatas td = new tempDatas();
48
49     private String filename;
50     private String taskC;
51
52     private String tempKeyboardtoType="file://" +
staticData.myDir+"/.src/Images/keyboard/tempToTypehtmlfile.html";
53     private String
tempKeyboard="file://" +staticData.myDir+"/.src/Images/keyboard/tempKeyboard.h
tml";
54
55     // public void closeFunction(MouseEvent mouseEvent) {
56     //
57     //     if(mouseEvent.getSource().equals(ele)){
58     //         bg.stop();
59     //         MainWindowController.stage.close();
60     //     }
61     // }
62     //
63     // @Override
64     // public void start(Stage primaryStage) throws Exception {
65     //     System.out.println("onStart");
66     // }
67     //
68     //
69     // @Override
70     // public void init() throws Exception {
71     //     super.init();
72     //     System.out.println("onInit");
73     // }
74     //
75     // @Override

```

```

76 //      public void stop() throws Exception {
77 //          System.out.println("onStop");
78 //
79 //          super.stop();
80 //      }
81
82 //      public class Backgrounds{
83 //
84 //
85 //          final Timeline timeline=new Timeline(
86 //              new KeyFrame(
87 //                  Duration.seconds(1),
88 //                  event -> {
89 //                      System.out.println("hello");
90 //                  }
91 //              )
92 //          );
93 //
94 //          public void go(){
95 //              timeline.setCycleCount(Animation.INDEFINITE);
96 //              timeline.play();
97 //          }
98 //          public void stop(){
99 //              timeline.stop();
100 //          }
101 //
102 //      }
103 //
104 //
105
106
107 //      public Backgrounds bg= new Backgrounds();
108
109      public void initialize() throws IOException {
110
111
112
113          MainWindowController.stage.close();
114
115
116      //          bg.go();
117
118
119          staticData.tempSwitch=false;
120          staticData.timerSeconds=0;
121
122
123
124
125          td.cursorPointer=0;
126          userNameLabel.setText(staticData.username);
127
128
129          td.t = new Thread(td.hw);
130          td.t1= new Thread(td.tw);
131
132          td.addLessons();

```



```

133         userNameLabel.setText(staticData.usrname);
134
135         Random random=new Random();
136
137         filename = staticData.myDir+"/.src/lesson/" +
td.lsnlistMap.get(random.nextInt(td.lsnlistMap.size()));
138
139         fileReaderClass f = new fileReaderClass(filename);
140         td.lsn = f.run();
141         // td.toPrint="";
142         lsnparts = td.lsn.split("");
143
144         lsnparts1=td.lsn.split("");
145         ptr = 0;
146         String strFirst = "";
147
148         String strWord = "";
149         while (!lsnparts[ptr].equals(" ")) {
150             strWord = strWord + lsnparts[ptr];
151             ptr++;
152             ////System.out.println(ptr);
153         }
154         //         System.out.println("1 :"+strWord);
155
156         String strLast = "";
157         boolean wordptr = false;
158         for (int i = ptr; i < 400; i++) {
159             ////System.out.println(i);
160             if (i % 85 == 0 && i > 0) {
161                 wordptr = true;
162             }
163             if (wordptr == true && lsnparts[i].equals(" ")) {
164                 wordptr = false;
165             }
166
167             strLast = strLast + lsnparts[i];
168             if (i == 399) {
169                 wordptr = true;
170                 for (int j = i + 1; j < 500; j++) {
171                     if (wordptr == true && lsnparts[j].equals(" ")) {
172                         wordptr = false;
173                         break;
174                     }
175                     strLast = strLast + lsnparts[j];
176                 }
177             }
178         }
179         td.tw.write(strFirst, strWord, strLast);
180         td.tl.run();
181
182         taskC="_";
183         //         userTyping.setText("");
184
185
186         td.toPrint="";
187         lsnparts= td.lsn.split("");
188         for(int i = 0 ;i<20;i++){

```

```

189         td.toPrint=td.toPrint+lsnparts[i];
190     }
191     td.tempString="";
192     for(int i = 1 ;i<20;i++){
193         td.tempString=td.tempString+lsnparts[i];
194     }
195     td.temp1=td.toPrint.split("");
196     keys.clear();
197     keys.put(td.temp1[td.cursorPointer],1);
198     td.hw.write(keys,taskC);
199     td.t.run();
200
201     keyboardLoad();
202
203
204 }
205
206 private void keyboardLoad() {
207
208
209
210
211     WebEngine webEngine= webopen.getEngine();
212     webEngine.load(tempKeyboard);
213
214     WebEngine webEngine1= toTypeViewLayoutWebView.getEngine();
215     webEngine1.load(tempKeyboardtoType);
216 }
217
218 public void onHandleKeyPressed(KeyEvent keyEvent) throws IOException
219 {
220     int x;
221     try {
222         x= (int) keyEvent.getText().charAt(0);
223     }catch (Exception e){
224         keys.clear();
225         keys.put(keyEvent.getCode().toString(),2);
226         keys.put(td.temp1[0],1);
227         td.hw.write(keys,taskC);
228         td.t.run();
229         keyboardLoad();
230         return;
231     }
232
233     if(x==8
234 ||x==127||x==16|| (keyEvent.getText().equals("CAPS")) || (keyEvent.getText().equ
235 als("ESCAPE")) )
236     {
237         keys.clear();
238         keys.put(keyEvent.getCode().toString(),2);
239         keys.put(td.temp1[td.cursorPointer],1);
240         td.hw.write(keys,taskC);
241         td.t.run();
242         keyboardLoad();

```

```

243         return;
244     }
245
246     lsnparts= td.lsn.split("");
247
248
249     String tempStr;
250     tempStr=keyEvent.getText();
251
252     if(keyEvent.isShiftDown()){
253         tempStr=td.is.convert(tempStr);
254     }
255
256
257     if(tempStr.equals(lsnparts[td.cursorPointer])){
258         if(!staticData.tempSwitch){
259             staticData.tempSwitch=true;
260         }
261
262         if(lsnparts[td.cursorPointer].equals(" ")){
263             td.spaceCounter++;
264             tempSpaceCounter++;
265             taskC="_";
266             //         userTyping.setText("");
267             tempcursorpointer=0;
268
269
270             int temp=0;
271             ptr=0;
272             String strFirst="";
273             while(temp!=tempSpaceCounter){
274                 if(lsnparts1[ptr].equals(" ")){
275                     temp++;
276                 }
277                 strFirst=strFirst+lsnparts1[ptr];
278                 ptr++;
279             }
280
281             tempptr=ptr-1;
282
283
284             String strWord = "";
285             while (!lsnparts1[ptr].equals(" ")) {
286                 strWord = strWord + lsnparts1[ptr];
287                 ptr++;
288             }
289
290
291             //         System.out.println("2 :"+strWord);
292
293             String strLast = "";
294             boolean wordptr = false;
295             for (int i = ptr; i < 400; i++) {
296                 if (i % 85 == 0 && i > 0) {
297                     wordptr = true;
298                 }
299                 if (wordptr == true && lsnparts1[i].equals(" ")) {

```

```

300         wordptr = false;
301     }
302     strLast = strLast + lsnparts1[i];
303     if (i == 399) {
304         wordptr = true;
305         for (int j = i + 1; j < 500; j++) {
306             if (wordptr == true && lsnparts1[j].equals("
307         )) {
308             wordptr = false;
309             break;
310         }
311         strLast = strLast + lsnparts1[j];
312     }
313 }
314 td.tw.write(strFirst, strWord, strLast);
315 td.tl.run();
316
317 }
318
319
320 td.toPrint="";
321 if(td.cursorPointer==td.lsn.length()){
322     td.lsnNext();
323 }
324
325
326 td.cursorPointer++;
327 tempcursorpointer++;
328
329
330 td.toPrint="";
331 for(int i=tempptr;i<tempptr+tempcursorpointer;i++){
332     td.toPrint=td.toPrint+lsnparts1[i];
333 }
334 taskC=td.toPrint+"_";
335 //     userTyping.setText(td.toPrint);
336
337 if(td.cursorPointer%400==0) {
338     tempSpaceCounter=0;
339     td.lstcursorPointer=td.cursorPointer;
340     int temp=td.cursorPointer;
341
342     while (!lsnparts[temp].equals(" ")){
343         temp--;
344     }
345
346     td.toPrint="";
347     temp++;
348     for (int i = temp; i < td.cursorPointer + 500; i++) {
349         td.toPrint = td.toPrint + lsnparts[i];
350     }
351
352
353
354     lsnparts1 = td.toPrint.split("");
355     ptr = 0;

```

```

356         String strFirst = "";
357
358         tempPtr=ptr;
359         String strWord = "";
360         while (!lsnparts1[ptr].equals(" ")) {
361             strWord = strWord + lsnparts1[ptr];
362             ptr++;
363         }
364
365
366
367         //          System.out.println("3 :"+strWord);
368
369         String strLast = "";
370         boolean wordPtr = false;
371         for (int i = ptr; i < 400; i++) {
372             if (i % 85 == 0 && i > 0) {
373                 wordPtr = true;
374             }
375             if (wordPtr == true && lsnparts1[i].equals(" ")) {
376                 wordPtr = false;
377             }
378             strLast = strLast + lsnparts1[i];
379             if (i == 399) {
380                 wordPtr = true;
381                 for (int j = i + 1; j < 500; j++) {
382                     if (wordPtr == true && lsnparts1[j].equals("
")) {
383                         wordPtr = false;
384                         break;
385                     }
386                     strLast = strLast + lsnparts1[j];
387                 }
388             }
389         }
390         td.tw.write(strFirst, strWord, strLast);
391         td.tl.run();
392
393
394
395
396
397     }
398
399
400
401
402
403
404
405
406         td.toPrint= "";
407         for (int i = td.cursorPointer; i < td.lstcursorPointer+400;
408 i++) {
409             td.toPrint = td.toPrint + lsnparts[i];
410

```

```

411         td.temp1=td.toPrint.split("");
412
413         if(td.temp1[0].equals(" ")){
414             td.temp1[0]="SPACE";
415         }
416
417
418
419         keys.clear();
420         keys.put(tempStr,0);
421         keys.put(td.temp1[0],1);
422
423         td.hw.write(keys,taskC);
424         td.t.run();
425
426         int temp=td.cursorPointer;
427
428
429         keyboardLoad();
430
431     }
432 }
433 else {
434     td.toPrint= "";
435     for (int i = td.lstcursorPointer; i < td.cursorPointer; i++)
436     {
437         td.toPrint = td.toPrint + lsnparts[i];
438     }
439     td.toPrint=td.toPrint+"_";
440
441     String temp;
442     temp= tempStr;
443
444     switch (keyEvent.getCode().toString()){
445         case "ESCAPE":
446             temp="ESCAPE";
447             break;
448         case "SPACE":
449             temp="SPACE";
450             break;
451         case "ENTER":
452             temp="ENTER";
453             break;
454         case "BACK_SPACE":
455             temp="BACK_SPACE";
456             break;
457         case "TAB":
458             temp="TAB";
459     }
460
461     keys.clear();
462     keys.put(temp,2);
463     keys.put(td.temp1[0],1);
464     td.hw.write(keys,taskC);
465     td.t.run();
466     keyboardLoad();

```

```

467
468         td.toPrint= "";
469         for (int i = td.cursorPointer+1; i < td.lstcursorPointer+20;
i++) {
470             td.toPrint = td.toPrint + lsnparts[i];
471         }
472     }
473     // System.out.println("CP :"+td.cursorPointer+" ptr: "+ptr+"
tptr:"+tempptr);
474 }
475
476
477 }
478

```

## StoriesTyping.fxml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.scene.control.Label?>
5  <?import javafx.scene.layout.BorderPane?>
6  <?import javafx.scene.layout.ColumnConstraints?>
7  <?import javafx.scene.layout.GridPane?>
8  <?import javafx.scene.layout.HBox?>
9  <?import javafx.scene.layout.RowConstraints?>
10 <?import javafx.scene.layout.VBox?>
11 <?import javafx.scene.paint.RadialGradient?>
12 <?import javafx.scene.paint.Stop?>
13 <?import javafx.scene.text.Font?>
14 <?import javafx.scene.web.WebView?>
15
16 <GridPane fx:id="MainGridPane" alignment="center" hgap="10"
onKeyPressed="#onHandleKeyPressed" style="-fx-background-color:white;"
vgap="10" xmlns="http://javafx.com/javafx/8.0.211"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdheshnayak.typingguru.StoriesTyping">
17   <VBox prefHeight="776.0" prefWidth="1410.0">
18     <children>
19       <BorderPane>
20         <center>
21           <HBox alignment="CENTER" prefHeight="90.0"
prefWidth="1179.0" BorderPane.alignment="CENTER">
22             <children>
23               <Label alignment="CENTER" text="Stories Typing">
24                 <textFill>
25                   <RadialGradient centerX="0.5" centerY="0.5"
radius="0.5">
26                     <stops>
27                       <Stop color="RED" />
28                       <Stop color="#610eae" offset="1.0" />
29                     </stops>
30                   </RadialGradient>
31                 </textFill>
32                 <font>
33                   <Font size="43.0" />

```

```

34         </font>
35     </Label>
36 </children>
37 </HBox>
38 </center>
39 <left>
40     <HBox alignment="CENTER" BorderPane.alignment="CENTER">
41         <children>
42             <Label fx:id="userNameLabel" text="Label" />
43         </children>
44         <BorderPane.margin>
45             <Insets bottom="10.0" left="10.0" right="10.0"
top="10.0" />
46         </BorderPane.margin>
47     </HBox>
48 </left>
49 <right>
50     <HBox alignment="CENTER" minWidth="150.0"
BorderPane.alignment="CENTER">
51         <children>
52             <Label text="" />
53             <Label fx:id="userSpeed" text="" />
54         </children>
55         <BorderPane.margin>
56             <Insets bottom="10.0" left="10.0" right="10.0"
top="10.0" />
57         </BorderPane.margin>
58     </HBox>
59 </right>
60 </BorderPane>
61 <BorderPane prefHeight="499.0" prefWidth="902.0"
VBox.vgrow="ALWAYS">
62     <bottom>
63         <WebView fx:id="webopen" maxWidth="960.0"
minHeight="460.0" prefHeight="399.0" prefWidth="600.0"
BorderPane.alignment="BOTTOM_CENTER" />
64     </bottom>
65     <top>
66         <WebView fx:id="toTypeViewLayoutWebView"
maxHeight="270.0" maxWidth="900.0" minHeight="250.0" minWidth="800.0"
prefHeight="270.0" prefWidth="-1.0" BorderPane.alignment="CENTER" />
67     </top>
68 </BorderPane>
69 </children>
70 </VBox>
71 <columnConstraints>
72     <ColumnConstraints />
73 </columnConstraints>
74 <rowConstraints>
75     <RowConstraints />
76 </rowConstraints>
77 </GridPane>
78

```



## topUsersView.java

```
1  package com.abdheshnayak.typingguru;
2
3  import com.abdheshnayak.typingguru.JavaClasses.staticData;
4  import javafx.event.ActionEvent;
5  import javafx.fxml.FXML;
6  import javafx.scene.control.*;
7  import javafx.scene.control.cell.PropertyValueFactory;
8  import javafx.stage.Stage;
9
10 import java.io.IOException;
11 import java.util.Map;
12
13 import static com.abdheshnayak.typingguru.JavaClasses.SorterClass.*;
14
15 public class topUsersView {
16
17     @FXML
18     private Label headline;
19     @FXML
20     private TableColumn tName;
21     @FXML
22     private TableColumn tScore;
23     @FXML
24     private Button closebtn;
25     @FXML
26     private TableView<data> tableview;
27     @FXML
28     private ListView viewScores;
29     @FXML
30     private ListView viewUsers;
31     public void initialize() throws IOException, ClassNotFoundException
32     {
33         tableview.setEditable(true);
34         tName.setCellValueFactory(new PropertyValueFactory<>("name"));
35         tScore.setCellValueFactory(new PropertyValueFactory<>("score"));
36
37         addUserClass user= new addUserClass();
38         user.readUsers();
39
40         Map<String, Integer> sortedMapAsc;
41         if(staticData.order==1) {
42             sortedMapAsc= sortByComparator(user.usrdata.Score, ASC);
43         }
44         else {
45             sortedMapAsc = sortByComparator(user.usrdata.Score, DESC);
46             headline.setText("Less Speed Users");
47         }
48
49         String[] keysName;
50         keysName = sortedMapAsc.keySet().toArray(new String[0]);
51
52         for (String i:keysName){
53             // System.out.println("hello");
```

```

54 //          System.out.println(i);
55          tableview.getItems().add(new
data(i,sortedMapAsc.get(i).toString()));
56
57      }
58  }
59
60
61
62      public void handleclosebtn(ActionEvent event) {
63          if(event.getSource().equals(closebtn)){
64              Stage stage= (Stage) closebtn.getScene().getWindow();
65              stage.close();
66          }
67      }
68
69      public static class data{
70          public String name;
71          public String score;
72
73          public data(String name, String score) {
74              this.name = name;
75              this.score = score;
76          }
77
78          public String getName() {
79              return name;
80          }
81
82          public String getScore() {
83              return score;
84          }
85      }
86  }

```

### UserSelector.java

```

1  package com.abdheshnayak.typingguru;
2
3  import com.abdheshnayak.typingguru.JavaClasses.staticData;
4  import javafx.beans.value.ChangeListener;
5  import javafx.beans.value.ObservableValue;
6  import javafx.event.EventHandler;
7  import javafx.fxml.FXML;
8  import javafx.fxml.FXMLLoader;
9  import javafx.scene.Scene;
10 import javafx.scene.control.*;
11 import javafx.scene.layout.BorderPane;
12 import javafx.scene.layout.GridPane;
13 import javafx.stage.Stage;
14 import javafx.stage.WindowEvent;
15
16 import java.io.IOException;
17 import java.util.*;

```

```

18
19 public class UserSelector {
20
21     public static String user;
22
23     @FXML
24     private Button btnStart;
25     @FXML
26     private Button btnDelete;
27     @FXML
28     private ChoiceBox choicebox;
29     @FXML
30     private Button btnCancel;
31     @FXML
32     private BorderPane UserSelectorBorderPane;
33     @FXML
34     private Button btnNewUser;
35
36     private addUserClass usr= new addUserClass();
37
38
39     private void initializeHelper() throws IOException,
ClassNotFoundException {
40
41         //         usr.WriteLanguage();
42         choicebox.getItems().clear();
43
44         choicebox.getItems().add("Guest User");
45         //         usr.writeUsers();
46         usr.readUsers();
47
48         String[] keysName= usr.usrdata.userNames.toArray(new String[0]);
49
50         choicebox.getItems().addAll(keysName);
51
52
53
54
55
56         choicebox.getSelectionModel().selectedIndexProperty().addListener(new
ChangeListener<Number>() {
57             @Override
58             public void changed(ObservableValue<? extends Number>
observable, Number oldValue, Number newValue) {
59
60                 }
61             });
62         choicebox.setValue("Guest User");
63     }
64
65     public void initialize() throws IOException, ClassNotFoundException
{
66
67         initializeHelper();
68
69

```

```

70     }
71
72     public void ButtonClickedHandle(javafx.scene.input.MouseEvent
mouseEvent) throws IOException, ClassNotFoundException {
73
74         if (mouseEvent.getSource().equals(btnNewUser)) {
75             userDialog("addUserWindow.fxml");
76             initializeHelper();
77         } else if (mouseEvent.getSource().equals(btnCancel)) {
78
79
80
81
82
83             Stage stage = (Stage) btnCancel.getScene().getWindow();
84             stage.close();
85             // System.out.println("Close Button Clicked");
86         } else if (mouseEvent.getSource().equals(btnStart)) {
87
88
89             if (choicebox.getValue().toString().equals("Guest User")) {
90                 staticData.username = choicebox.getValue().toString();
91                 Stage stage = (Stage) btnCancel.getScene().getWindow();
92                 stage.close();
93                 // Stage stage = new Stage();
94                 GridPane root;
95                 root = (GridPane)
FXMLLoader.load(getClass().getResource("Mainwindow.fxml"));
96                 // PasswordChecker ps = new PasswordChecker();
97                 // ps.usr= choicebox.getValue().toString();
98                 Scene scene = new Scene(root, 1280, 720);
99                 stage.setScene(scene);
100                 stage.show();
101
102             } else {
103
104                 staticData.username = choicebox.getValue().toString();
105                 Stage stage = (Stage) btnCancel.getScene().getWindow();
106                 stage.close();
107                 // Stage stage = new Stage();
108                 GridPane root;
109                 root = (GridPane)
FXMLLoader.load(getClass().getResource("passwordChecker.fxml"));
110                 // PasswordChecker ps = new PasswordChecker();
111                 // ps.usr= choicebox.getValue().toString();
112                 Scene scene = new Scene(root, 500, 200);
113                 stage.setScene(scene);
114                 stage.show();
115             }
116         } else if (mouseEvent.getSource().equals(btnDelete)&&
(!choicebox.getValue().toString().equals("Guest User"))) {
117             // System.out.println("Delete btn Clicked");
118             staticData.username = choicebox.getValue().toString();
119             Stage stage = new Stage();
120             GridPane root;
121             root = (GridPane)
FXMLLoader.load(getClass().getResource("deleteUser.fxml"));

```

```

122         //         PasswordChecker ps = new PasswordChecker();
123         //         ps.usr= choicebox.getValue().toString();
124         Scene scene = new Scene(root, 600, 300);
125         stage.setScene(scene);
126         stage.showAndWait();
127         initializeHelper();
128     }
129
130 }
131
132     public void userDialog(String fxmlName) throws IOException,
ClassNotFoundException {
133         Dialog<ButtonType> dialog = new Dialog<>();
134         dialog.initOwner(UserSelectorBorderPane.getScene().getWindow());
135         FXMLLoader fxmlLoader= new FXMLLoader();
136         fxmlLoader.setLocation(getClass().getResource(fxmlName));
137         try {
138             dialog.getDialogPane().setContent(fxmlLoader.load());
139         }catch (IOException e){
140             //         System.out.println("Exception:");
141             e.printStackTrace();
142             return;
143         }
144
145         //         Parent root =
FXMLLoader.load(getClass().getResource("addUserWindow.fxml"));
146         //         dialog.getDialogPane().setContent(root);
147         //         dialog.setTitle("Add New User");
148         dialog.getDialogPane().getButtonTypes().add(ButtonType.OK);
149         dialog.getDialogPane().getButtonTypes().add(ButtonType.CLOSE);
150         //         System.out.println("Ok");
151         Optional<ButtonType> result = dialog.showAndWait();
152         if(result.isPresent() && result.get() == ButtonType.OK) {
153             addUserClass newUser = fxmlLoader.getController();
154             if(newUser.save() == 1) {
155                 result=dialog.showAndWait();
156             }
157
158         //         System.out.println("OK Pressed");
159
160         }else {
161             //         System.out.println("Cancel Pressed");
162         }
163
164     }
165 }
166
167 }
168

```

### UserSelector.fxml

```

1     <?xml version="1.0" encoding="UTF-8"?>
2
3     <?import javafx.geometry.Insets?>
4     <?import javafx.scene.control.Button?>
5     <?import javafx.scene.control.ChoiceBox?>

```



```

52             <Insets right="30.0" />
53         </HBox.margin>
54         <font>
55             <Font size="22.0" />
56         </font>
57     </Label>
58     <ChoiceBox fx:id="choicebox" prefWidth="150.0">
59         <items>
60
61
62             </items>
63         </ChoiceBox>
64     </children>
65 </HBox>
66 </children>
67 </VBox>
68 </top>
69 <center>
70     <HBox alignment="BOTTOM_CENTER" prefHeight="100.0"
prefWidth="200.0" BorderPane.alignment="CENTER">
71         <children>
72             <Label alignment="BOTTOM_CENTER" contentDisplay="CENTER"
text="New User ??" textFill="#1f1263">
73                 <font>
74                     <Font name="Cousine Bold" size="18.0" />
75                 </font>
76             </Label>
77             <Button fx:id="btnNewUser" mnemonicParsing="false"
onMouseClicked="#ButtonClickedHandle" text="Add New User">
78                 <font>
79                     <Font size="15.0" />
80                 </font>
81                 <HBox.margin>
82                     <Insets left="25.0" />
83                 </HBox.margin>
84             </Button>
85         </children>
86     </HBox>
87 </center>
88 <bottom>
89     <BorderPane prefHeight="200.0" prefWidth="200.0"
BorderPane.alignment="CENTER">
90         <right>
91             <HBox alignment="BOTTOM_RIGHT" prefHeight="100.0"
prefWidth="200.0" BorderPane.alignment="CENTER">
92                 <children>
93                     <Button fx:id="btnCancel" minWidth="80.0"
mnemonicParsing="false" onMouseClicked="#ButtonClickedHandle" style="-fx-
font-size: 18;" text="Close">
94                         <HBox.margin>
95                             <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
96                         </HBox.margin>
97                     </Button>
98                     <Button fx:id="btnStart" minWidth="80.0"
mnemonicParsing="false" onMouseClicked="#ButtonClickedHandle" style="-fx-
font-size: 18;" text="Start">

```

```

99             <HBox.margin>
100                 <Insets bottom="10.0" left="10.0"
right="20.0" top="10.0" />
101             </HBox.margin>
102         </Button>
103     </children>
104 </HBox>
105 </right>
106 <left>
107     <HBox alignment="BOTTOM_LEFT" prefHeight="100.0"
prefWidth="200.0" BorderPane.alignment="BOTTOM_LEFT">
108         <children>
109             <Button fx:id="btnDelete" minWidth="80.0"
mnemonicParsing="false" onMouseClicked="#ButtonClickedHandle" style="-fx-
font-size: 18;" text="Delete" textFill="RED">
110                 <HBox.margin>
111                     <Insets bottom="10.0" left="10.0"
right="10.0" top="10.0" />
112                 </HBox.margin>
113             </Button>
114         </children>
115     </HBox>
116 </left>
117 </BorderPane>
118 </bottom>
119 </BorderPane>
120 <columnConstraints>
121     <ColumnConstraints />
122 </columnConstraints>
123 <rowConstraints>
124     <RowConstraints />
125 </rowConstraints>
126 </GridPane>
127

```

### topUsersView.fxml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.scene.control.Button?>
5  <?import javafx.scene.control.Label?>
6  <?import javafx.scene.control.TableColumn?>
7  <?import javafx.scene.control.TableView?>
8  <?import javafx.scene.layout.BorderPane?>
9  <?import javafx.scene.text.Font?>
10
11  <BorderPane prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/8.0.211" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdheshnayak.typingguru.topUsersView">
12      <top>
13          <Label fx:id="headline" text="Top Users" textFill="#370ada"
BorderPane.alignment="CENTER">
14              <font>
15                  <Font size="21.0" />
16              </font>
17          <BorderPane.margin>

```



```

18         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
19     </BorderPane.margin>
20 </Label>
21 </top>
22 <bottom>
23     <Button mnemonicParsing="false" text="Close" fx:id="closebtn"
onAction="#handleclosebtn" textFill="#a81313"
BorderPane.alignment="BOTTOM_RIGHT">
24         <BorderPane.margin>
25             <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
26         </BorderPane.margin>
27         <font>
28             <Font size="15.0" />
29         </font>
30     </Button>
31 </bottom>
32 <center>
33     <TableView fx:id="tableview" BorderPane.alignment="CENTER">
34         <columns>
35             <TableColumn fx:id="tName" maxWidth="4000.0" minWidth="-1.0"
prefWidth="451.0" text="Name" />
36             <TableColumn fx:id="tScore" maxWidth="258.0" minWidth="100.0"
prefWidth="127.0" text="Score" />
37         </columns>
38         <columnResizePolicy>
39             <TableView fx:constant="CONSTRAINED_RESIZE_POLICY" />
40         </columnResizePolicy>
41     <BorderPane.margin>
42         <Insets bottom="10.0" left="10.0" right="10.0" top="10.0" />
43     </BorderPane.margin>
44 </TableView>
45 </center>
46 </BorderPane>
47

```

### aboutApp.fxml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import javafx.geometry.Insets?>
4  <?import javafx.scene.control.Label?>
5  <?import javafx.scene.layout.ColumnConstraints?>
6  <?import javafx.scene.layout.GridPane?>
7  <?import javafx.scene.layout.HBox?>
8  <?import javafx.scene.layout.RowConstraints?>
9  <?import javafx.scene.layout.VBox?>
10 <?import javafx.scene.text.Font?>
11 <?import javafx.scene.web.WebView?>
12
13 <GridPane alignment="CENTER" prefHeight="400.0" prefWidth="600.0"
xmlns="http://javafx.com/javafx/8.0.211" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="com.abdheshnayak.typingguru.AboutApp">
14     <children>
15         <VBox prefHeight="200.0" prefWidth="100.0" GridPane.hgrow="ALWAYS"
GridPane.vgrow="ALWAYS">
16             <children>

```

```

17         <HBox alignment="CENTER" style="-fx-background-color: #000099;">
18             <children>
19                 <Label alignment="CENTER" contentDisplay="CENTER" text="About
App" textFill="white">
20                     <font>
21                         <Font name="System Bold" size="27.0" />
22                     </font>
23                     <HBox.margin>
24                         <Insets bottom="15.0" top="15.0" />
25                     </HBox.margin>
26                 </Label>
27             </children>
28         </HBox>
29         <WebView fx:id="aboutUsPage" prefHeight="200.0" prefWidth="200.0"
VBox.vgrow="ALWAYS" />
30     </children>
31 </VBox>
32 </children>
33 <columnConstraints>
34     <ColumnConstraints />
35 </columnConstraints>
36 <rowConstraints>
37     <RowConstraints />
38 </rowConstraints>
39 </GridPane>
40

```

### htmlWriter.java

```

1  package com.abdheshnayak.typingguru.keyboardController;
2
3  import com.abdheshnayak.typingguru.JavaClasses.staticData;
4
5  import java.io.FileNotFoundException;
6  import java.io.FileReader;
7  import java.io.FileWriter;
8  import java.io.IOException;
9  import java.util.ArrayList;
10 import java.util.HashMap;
11 import java.util.Map;
12
13 public class htmlWriter implements Runnable {
14
15
16     private Map<String, Integer> capital = new HashMap<String,
Integer>();
17     private Map<String, Integer> All = new HashMap<String, Integer>();
18     private ArrayList<String> datas = new ArrayList<String>();
19     private String pname=
staticData.myDir+"/.src/Images/keyboard/tempKeyboard.html";
20
21     public void ChangeLanguage(String language){
22         datas.set(All.get("Language"), "\""+language+".css\"");
23     }
24

```

```

25     public htmlWriter() {
26
27
28         datas.add("style=\"background-color:green;box-shadow:0px 0px
10px green;\"");
29         All.put("green", 0);
30         datas.add("style=\"background-color:red;box-shadow:0px 0px 10px
pink;\"");
31         All.put("red", All.size());
32         datas.add("<!DOCTYPE html PUBLIC \"-//W3C//DTD XHTML 1.0
Strict//EN\" \"https://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd\">\n" +
33             "<html xmlns=\"http://www.w3.org/1999/xhtml\"
xml:lang=\"en\" lang=\"en\">\n" +
34             "<head>\n" +
35             "    <meta http-equiv=\"Content-Type\"
content=\"text/html; charset=UTF-8\" />\n" +
36             "    <meta name=\"viewport\" content=\"width=device-
width, initial-scale=1.0\">\n" +
37             "\n" +
38             "    <title>Online Keyboard</title>\n" +
39             "    <link rel=\"stylesheet\" type=\"text/css\" href=");
40         All.put("htmlStart",All.size());
41         if (StaticData.keyBoardType){
42             datas.add("\"English.css\"");
43             All.put("Language",All.size());
44         }else {
45             datas.add("\"Nepali.css\"");
46             All.put("Language",All.size());
47         }
48         datas.add("</head>\n" +
49             "<body>\n" +
50             "\n" +
51             "<div style=\"display: flex; flex-direction: column;
align-items: center;\">\n" +
52             "    <div class=\"FontSize\";>\n" +
53             "        <div style=\"padding: 0px; color:blue;\">");
54
55
56         All.put("htmlStart1",All.size());
57         datas.add("As sa asd dsa asdf fdsa sdf lk");
58         All.put("ToType",All.size());
59         datas.add("</div>\n" +
60             "        <div style=\"padding: 0px; height: 5px; padding-
bottom:30px;\">\n");
61         All.put("htmlStart2",All.size());
62         datas.add("        <!-- react-text: 306 -->A<!-- /react-text --
><!-- react-text: 310 -->s\n" +
63             "        <!-- /react-text --><!-- react-text: 311 -->\n" +
64             "        <!-- /react-text --><!-- react-text: 312 -->s\n"
+
65             "        <!-- /react-text -->\n" );
66
67         All.put("UserTyping",All.size());
68         datas.add("    </div>\n" +
69             "    </div>\n"+
70             "<div class=\"keyboard\" >\n" +
71             "\n" +

```

```

72         "        <div class=\"section-a\">\n" +
73         "        <div class=\"key num dual\">";
74 All.put("htmlStart3", All.size());
75 //      datas.add(">\n" +
76 //      "        Esc\n" +
77 //      "        </div>\n" +
78 //      "\n" +
79 //      "        <div class=\"key function\">;
80 // All.put("ESCAPE", All.size());
81 //      datas.add(">\n" +
82 //      "        F1\n" +
83 //      "        </div>\n" +
84 //      "        <div class=\"key function\">;
85 // All.put("F1", All.size());
86 //      datas.add(">\n" +
87 //      "        F2\n" +
88 //      "        </div>\n" +
89 //      "        <div class=\"key function\">;
90 // All.put("F2", All.size());
91 //      datas.add(">\n" +
92 //      "        F3\n" +
93 //      "        </div>\n" +
94 //      "\n" +
95 //      "        <div class=\"key function space2\">;
96 // All.put("F3", All.size());
97 //      datas.add(">\n" +
98 //      "        F4\n" +
99 //      "        </div>\n" +
100 //      "        <div class=\"key function\">;
101 // All.put("F4", All.size());
102 //      datas.add(">\n" +
103 //      "        F5\n" +
104 //      "        </div>\n" +
105 //      "        <div class=\"key function\">;
106 // All.put("F5", All.size());
107 //      datas.add(">\n" +
108 //      "        F6\n" +
109 //      "        </div>\n" +
110 //      "        <div class=\"key function\">;
111 // All.put("F6", All.size());
112 //      datas.add(">\n" +
113 //      "        F7\n" +
114 //      "        </div>\n" +
115 //      "        <div class=\"key function space2\">;
116 // All.put("F7", All.size());
117 //      datas.add(">\n" +
118 //      "        F8\n" +
119 //      "        </div>\n" +
120 //      "\n" +
121 //      "        <div class=\"key function\">;
122 // All.put("F8", All.size());
123 //      datas.add(">\n" +
124 //      "        F9\n" +
125 //      "        </div>\n" +
126 //      "        <div class=\"key function\">;
127 // All.put("F9", All.size());
128 //      datas.add(">\n" +

```

```

129 //          "          F10\n" +
130 //          "          </div>\n" +
131 //          "          <div class=\"key function\">;
132 //      All.put("F10", All.size());
133 //      datas.add(">\n" +
134 //          "          F11\n" +
135 //          "          </div>\n" +
136 //          "          <div class=\"key function\">;
137 //      All.put("F11", All.size());
138 //      datas.add(">\n" +
139 //          "          F12\n" +
140 //          "          </div>\n" +
141 //          "          <!--END FUNCTION KEYS -->\n" +
142 //          "\n" +
143 //          "          <div class=\"key num dual\">;
144 //      All.put("F12", All.size());
145 //      datas.add(">\n" +
146 //          "          ~<br>&nbsp;~\n" +
147 //          "          </div>\n" +
148 //          "\n" +
149 //          "          <div class=\"key num dual\">;
150
151 //      capital.put("~", All.size());
152 //      All.put("`", All.size());
153 //      datas.add(">\n" +
154 //          "          !<br>&nbsp;1\n" +
155 //          "          </div>\n" +
156 //          "          <div class=\"key num dual\">;
157
158 //      capital.put("!", All.size());
159 //      All.put("1", All.size());
160 //      datas.add(">\n" +
161 //          "          @<br>&nbsp;2\n" +
162 //          "          </div>\n" +
163 //          "          <div class=\"key num dual\">;
164
165 //      capital.put("@", All.size());
166 //      All.put("2", All.size());
167 //      datas.add(">\n" +
168 //          "          #<br>&nbsp;3\n" +
169 //          "          </div>\n" +
170 //          "          <div class=\"key num dual\">;
171 //      capital.put("#", All.size());
172 //      All.put("3", All.size());
173 //      datas.add(">\n" +
174 //          "          $<br>&nbsp;4\n" +
175 //          "          </div>\n" +
176 //          "          <div class=\"key num dual\">;
177 //      capital.put("$", All.size());
178 //      All.put("4", All.size());
179 //      datas.add(">\n" +
180 //          "          %<br>&nbsp;5\n" +
181 //          "          </div>\n" +
182 //          "          <div class=\"key num dual\">;
183 //      capital.put("%", All.size());
184 //      All.put("5", All.size());
185 //      datas.add(">\n" +

```

```

186         ^<br>&nbsp;6\n" +
187         "</div>\n" +
188         "<div class=\"key num dual\">";
189 capital.put("^", All.size());
190 All.put("6", All.size());
191 datas.add(">\n" +
192         "&<br>&nbsp;7\n" +
193         "</div>\n" +
194         "<div class=\"key num dual\">";
195 capital.put("&", All.size());
196 All.put("7", All.size());
197 datas.add(">\n" +
198         "*<br>&nbsp;8\n" +
199         "</div>\n" +
200         "<div class=\"key num dual\">";
201 capital.put("*", All.size());
202 All.put("8", All.size());
203 datas.add(">\n" +
204         "(<br>&nbsp;9\n" +
205         "</div>\n" +
206         "<div class=\"key num dual\">";
207 capital.put("(", All.size());
208 All.put("9", All.size());
209 datas.add(">\n" +
210         ")<br>&nbsp;0\n" +
211         "</div>\n" +
212         "<div class=\"key num dual\">";
213 capital.put(")", All.size());
214 All.put("0", All.size());
215 datas.add(">\n" +
216         "_<br>&nbsp;- \n" +
217         "</div>\n" +
218         "<div class=\"key num dual\">";
219 capital.put("_", All.size());
220 All.put("-", All.size());
221 datas.add(">\n" +
222         "+<br>&nbsp;= \n" +
223         "</div>\n" +
224         "<div class=\"key backspace\">";
225 capital.put("+", All.size());
226 All.put "=", All.size());
227 datas.add(">\n" +
228         "Backspace\n" +
229         "</div>\n" +
230         "<!--END NUMBER KEYS -->\n" +
231         "\n" +
232         "<div class=\"key tab\">";
233 All.put("BACK_SPACE", All.size());
234 datas.add(">\n" +
235         "Tab\n" +
236         "</div>\n" +
237         "\n" +
238         "<div class=\"key dual\">";
239 All.put("TAB", All.size());
240 datas.add(">\n" +
241         "Q<br> &nbsp;q" +
242         "</div>\n" +

```

```

243         "           <div class=\"key dual\">;
244
245 capital.put("Q", All.size());
246 All.put("q", All.size());
247 datas.add(">\n" +
248         "           W\n<br> &nbsp;w" +
249         "           </div>\n" +
250         "           <div class=\"key dual\">;
251 capital.put("W", All.size());
252 All.put("w", All.size());
253 datas.add(">\n" +
254         "           E\n<br> &nbsp;e" +
255         "           </div>\n" +
256         "           <div class=\"key dual\">;
257 capital.put("E", All.size());
258 All.put("e", All.size());
259 datas.add(">\n" +
260         "           R<br> &nbsp;r\n" +
261         "           </div>\n" +
262         "           <div class=\"key dual\">;
263 capital.put("R", All.size());
264 All.put("r", All.size());
265 datas.add(">\n" +
266         "           T<br> &nbsp;t\n" +
267         "           </div>\n" +
268         "           <div class=\"key dual\">;
269 capital.put("T", All.size());
270 All.put("t", All.size());
271 datas.add(">\n" +
272         "           Y<br> &nbsp;y\n" +
273         "           </div>\n" +
274         "           <div class=\"key dual\">;
275 capital.put("Y", All.size());
276 All.put("y", All.size());
277 datas.add(">\n" +
278         "           U<br> &nbsp;u\n" +
279         "           </div>\n" +
280         "           <div class=\"key dual\">;
281 capital.put("U", All.size());
282 All.put("u", All.size());
283 datas.add(">\n" +
284         "           I<br> &nbsp;i\n" +
285         "           </div>\n" +
286         "           <div class=\"key dual\">;
287 capital.put("I", All.size());
288 All.put("i", All.size());
289 datas.add(">\n" +
290         "           O<br> &nbsp;o\n" +
291         "           </div>\n" +
292         "           <div class=\"key dual\">;
293 capital.put("O", All.size());
294 All.put("o", All.size());
295 datas.add(">\n" +
296         "           P<br> &nbsp;p\n" +
297         "           </div>\n" +
298         "           <div class=\"key dual\">;
299 capital.put("P", All.size());

```

```

300 All.put("p", All.size());
301 datas.add(">\n" +
302     "        {<Br>&nbsp;[ \n" +
303     "        </div>\n" +
304     "        <div class=\"key dual\"");
305 capital.put("{", All.size());
306 All.put("[", All.size());
307 datas.add(">\n" +
308     "        }<br>&nbsp;] \n" +
309     "        </div>\n" +
310     "        <div class=\"key dual dual slash\"");
311 capital.put("}", All.size());
312 All.put("]", All.size());
313 datas.add(">\n" +
314     "        |<br>&nbsp;\\ \n" +
315     "        </div>\n" +
316     "        <div class=\"key caps\"");
317 capital.put("|", All.size());
318 All.put("\\", All.size());
319 datas.add(">\n" +
320     "        Caps<br>Lock\n" +
321     "        </div>\n" +
322     "        <div class=\"key dual\"");
323 All.put("CAPS", All.size());
324 datas.add(">\n" +
325     "        A<br> &nbsp;a\n" +
326     "        </div>\n" +
327     "        <div class=\"key dual\"");
328 capital.put("A", All.size());
329 All.put("a", All.size());
330 datas.add(">\n" +
331     "        S<br> &nbsp;s\n" +
332     "        </div>\n" +
333     "        <div class=\"key dual\"");
334 capital.put("S", All.size());
335 All.put("s", All.size());
336 datas.add(">\n" +
337     "        D<br> &nbsp;d\n" +
338     "        </div>\n" +
339     "        <div class=\"key dual\"");
340 capital.put("D", All.size());
341 All.put("d", All.size());
342 datas.add(">\n" +
343     "        F<br> &nbsp;f\n" +
344     "        </div>\n" +
345     "        <div class=\"key dual\"");
346 capital.put("F", All.size());
347 All.put("f", All.size());
348 datas.add(">\n" +
349     "        G<br> &nbsp;g\n" +
350     "        </div>\n" +
351     "        <div class=\"key dual\"");
352 capital.put("G", All.size());
353 All.put("g", All.size());
354 datas.add(">\n" +
355     "        H<br> &nbsp;h\n" +
356     "        </div>\n" +

```



```

357         "           <div class=\"key dual\">;
358 capital.put("H", All.size());
359 All.put("h", All.size());
360 datas.add(">\n" +
361         "           J<br> &nbsp;j\n" +
362         "           </div>\n" +
363         "           <div class=\"key dual\">;
364 capital.put("J", All.size());
365 All.put("j", All.size());
366 datas.add(">\n" +
367         "           K<br> &nbsp;k\n" +
368         "           </div>\n" +
369         "           <div class=\"key dual\">;
370 capital.put("K", All.size());
371 All.put("k", All.size());
372 datas.add(">\n" +
373         "           L<br> &nbsp;l\n" +
374         "           </div>\n" +
375         "           <div class=\"key dual\">;
376 capital.put("L", All.size());
377 All.put("l", All.size());
378 datas.add(">\n" +
379         "           :<br>&nbsp;;\n" +
380         "           </div>\n" +
381         "           <div class=\"key dual\">;
382 capital.put(":", All.size());
383 All.put(";", All.size());
384 datas.add(">\n" +
385         "           \"<br>&nbsp;'\n" +
386         "           </div>\n" +
387         "           <div class=\"key enter\">;
388 capital.put("\"", All.size());
389 All.put("'", All.size());
390 datas.add(">\n" +
391         "           Enter\n" +
392         "           </div>\n" +
393         "\"\n" +
394         "           <div class=\"key shift left\">;
395 All.put("ENTER", All.size());
396 datas.add(">\n" +
397         "           Shift\n" +
398         "           </div>\n" +
399         "           <div class=\"key dual\">;
400 All.put("SHIFT", All.size());
401 datas.add(">\n" +
402         "           Z<br> &nbsp;z\n" +
403         "           </div>\n" +
404         "           <div class=\"key dual\">;
405 capital.put("Z", All.size());
406 All.put("z", All.size());
407 datas.add(">\n" +
408         "           X<br> &nbsp;x\n" +
409         "           </div>\n" +
410         "           <div class=\"key dual\">;
411 capital.put("X", All.size());
412 All.put("x", All.size());
413 datas.add(">\n" +

```

```

414         "            C<br> &nbsp;c\n" +
415         "            </div>\n" +
416         "            <div class=\"key dual\">;
417 capital.put("C", All.size());
418 All.put("c", All.size());
419 datas.add(">\n" +
420         "            V<br> &nbsp;v\n" +
421         "            </div><div class=\"key dual\">;
422 capital.put("V", All.size());
423 All.put("v", All.size());
424 datas.add(">\n" +
425         "            B<br> &nbsp;b\n" +
426         "            </div><div class=\"key dual\">;
427 capital.put("B", All.size());
428 All.put("b", All.size());
429 datas.add(">\n" +
430         "            N<br> &nbsp;n\n" +
431         "            </div><div class=\"key dual\">;
432 capital.put("N", All.size());
433 All.put("n", All.size());
434 datas.add(">\n" +
435         "            M<br> &nbsp;m\n" +
436         "            </div>\n" +
437         "            <div class=\"key dual\">;
438 capital.put("M", All.size());
439 All.put("m", All.size());
440 datas.add(">\n" +
441         "            < <br>&nbsp;, \n" +
442         "            </div>\n" +
443         "            <div class=\"key dual\">;
444 capital.put("<", All.size());
445 All.put(",", All.size());
446 datas.add(">\n" +
447         "            ><br>&nbsp;.\n" +
448         "            </div>\n" +
449         "            <div class=\"key dual\">;
450 capital.put(">", All.size());
451 All.put(".", All.size());
452 datas.add(">\n" +
453         "            ?<br>&nbsp;/\n" +
454         "            </div>\n" +
455         "            <div class=\"key shift right\">;
456 capital.put("?", All.size());
457 All.put("/", All.size());
458 datas.add(">\n" +
459         "            Shift\n" +
460         "            </div>\n" +
461         "            <div class=\"key ctrl\">;
462 All.put("SHIFT1", All.size());
463 datas.add(">\n" +
464         "            Ctrl\n" +
465         "            </div>\n" +
466         "\n" +
467         "            <div class=\"key\">;
468 All.put("CONTROL", All.size());
469 datas.add(">\n" +
470         "            Win\n" +

```

```

471         "        </div>\n" +
472         "        <div class=\"key\">";
473 All.put("WIN", All.size());
474 datas.add(">\n" +
475         "        Alt\n" +
476         "        </div>\n" +
477         "        <div class=\"key space\">";
478 All.put("ALT", All.size());
479 datas.add(">\n" +
480         "\n" +
481         "\n" +
482         "\n" +
483         "        </div>\n" +
484         "        <div class=\"key\">";
485 All.put("SPACE", All.size());
486 datas.add(">\n" +
487         "        Alt\n" +
488         "        </div>\n" +
489         "        <div class=\"key\">";
490 All.put("ALT1", All.size());
491 datas.add(">\n" +
492         "        Win\n" +
493         "        </div>\n" +
494         "        <div class=\"key\">";
495 All.put("WIN1", All.size());
496 datas.add(">\n" +
497         "        Fn\n" +
498         "        </div>\n" +
499         "\n" +
500         "\n" +
501         "        <div class=\"key ctrl\">";
502 All.put("FN", All.size());
503 datas.add(">\n" +
504         "        Ctrl\n" +
505         "        </div>\n");
506 All.put("CONTROL1", All.size());
507 datas.add("\n" +
508         "    </div><!-- end section-a-->\n" +
509         "\n" +
510         "</div><!-- end sec-func -->\n" +
511         "\n" +
512         "\n" +
513         "</div><!-- end section-b-->\n" +
514         "\n" +
515         "</div>\n" +
516         "</body>\n" +
517         "</html>");
518 All.put("endHtml", All.size());
519
520 }
521
522 String lsn = new String();
523
524 public String read(String pname) throws IOException {
525     lsn = "";
526     FileReader inputStream = null;
527     int x = 0;

```

```

528         try {
529             inputStream = new FileReader(String.valueOf(pname));
530             int c;
531             while ((c = inputStream.read()) != -1) {
532                 lsn = lsn + String.valueOf((char) c);
533             }
534         } catch (FileNotFoundException e) {
535             e.printStackTrace();
536         } catch (IOException e) {
537             e.printStackTrace();
538         } finally {
539             if (inputStream != null) {
540                 inputStream.close();
541             }
542         }
543         //      System.out.println(lsn);
544         //      System.out.println("Read");
545         return lsn;
546     }
547
548     private Map<String,Integer> keys;
549
550
551     public String write(Map<String, Integer> KeysList,String ToType,
String UserTyping) throws IOException {
552         keys=KeysList;
553         String temp="";
554         for (int i=0;i<ToType.length();i++){
555             if(ToType.charAt(i)==' ') {
556                 temp=temp+"&nbsp;";
557             }else {
558                 temp+=ToType.charAt(i);
559             }
560         }
561         datas.set(All.get("ToType"),temp);
562         temp="";
563         for (int i=0;i<UserTyping.length();i++){
564             if(UserTyping.charAt(i)==' ') {
565                 temp=temp+"&nbsp;";
566             }else if(UserTyping.charAt(i)=='_'){
567                 temp+="<span style=\"font-family:arial;\">_</span>";
568             }else {
569                 temp+=UserTyping.charAt(i);
570             }
571         }
572         //      System.out.println(temp);
573         datas.set(All.get("UserTyping"),temp);
574         return lsn;
575     }
576
577
578     public String write(Map<String, Integer> KeysList, String
UserTyping) throws IOException {
579         keys=KeysList;
580         datas.set(All.get("ToType"), " ");
581
582         String temp="";

```

```

583         for (int i=0;i<UserTyping.length();i++){
584             if(UserTyping.charAt(i)==' ') {
585                 temp=temp+"&nbsp;";
586             }else {
587                 temp+=UserTyping.charAt(i);
588             }
589         }
590         datas.set(All.get("UserTyping"),temp);
591         datas.set(All.get("UserTyping"),UserTyping);
592         return lsn;
593     }
594
595     public String write(Map<String, Integer> KeysList) throws
IOException {
596         keys=KeysList;
597         return lsn;
598     }
599
600     public void setToNull(){
601         datas.set(All.get("ToType")," ");
602         datas.set(All.get("UserTyping")," ");
603     }
604
605
606     @Override
607     public void run(){
608         FileWriter outputStream = null;
609         int x = 0;
610         lsn = "";
611         if(keys.containsKey("ALT_GRAPH")){
612             keys.put("ALT",keys.get("ALT_GRAPH"));
613             keys.remove("ALT_GRAPH");
614         }
615         if(keys.containsKey(" ")){
616             keys.put("SPACE",keys.get((" ")));
617             keys.remove(" ");
618         }
619         String[] keysName = keys.keySet().toArray(new String[0]);
620         //      System.out.println(keys.entrySet());
621         for (int i = (int) All.get("htmlStart"); i < All.size(); i++) {
622             for (String s : keysName) {
623
624                 if (All.containsKey(s)) {
625                     if ((i == All.get(s)) && (keys.get(s) == 1)) {
626                         lsn = lsn + datas.get(All.get("green"));
627
628                         if (s == "CONTROL") {
629                             keys.put("CONTROL1",1);
630                         } else if (s == "SHIFT") {
631                             keys.put("SHIFT1",1);
632                         } else if (s == "ALT") {
633                             keys.put("ALT1",1);
634                         } else if (s == "WIN") {
635                             keys.put("WIN1",1);
636                         } else {
637

```

```

638         } else if ((i == All.get(s)) && (keys.get(s) == 2))
639         {
640             lsn = lsn + datas.get(All.get("red"));
641
642             if (s == "CONTROL") {
643                 keys.put("CONTROL1",2);
644             } else if (s == "SHIFT") {
645                 keys.put("SHIFT1",2);
646             } else if (s == "ALT") {
647                 keys.put("ALT1",2);
648             } else if (s == "WIN") {
649                 keys.put("WIN1",2);
650             } else {
651             }
652         } else if (capital.containsKey(s) && keys.get(s) != 0) {
653             // System.out.println("Found");
654             keys.put("SHIFT",keys.get(s));
655             if ((i == capital.get(s)) && (keys.get(s) == 1)) {
656                 lsn = lsn + datas.get(All.get("green"));
657             } else if ((i == capital.get(s)) && (keys.get(s) ==
658 2)) {
659                 lsn = lsn + datas.get(All.get("red"));
660             }
661         }
662
663         keysName = keys.keySet().toArray(new String[0]);
664         // System.out.println(keys.entrySet());
665     }
666     lsn = lsn + datas.get(i);
667
668     try {
669         outputStream = new FileWriter(String.valueOf(pname));
670         outputStream.write(lsn);
671     } catch (FileNotFoundException e) {
672         e.printStackTrace();
673     } catch (IOException e) {
674         e.printStackTrace();
675     } finally {
676         if (outputStream != null) {
677             try {
678                 outputStream.close();
679             } catch (IOException e) {
680                 e.printStackTrace();
681             }
682         }
683         try {
684             outputStream.close();
685         } catch (IOException e) {
686             e.printStackTrace();
687         }
688     }
689 }
690 // System.out.println(lsn);
691 // System.out.println("Written");
692

```

```
693
694     }
695 }
696
```

### fileReaderClass.java

```
1  package com.abdreshnayak.typingguru.JavaClasses;
2
3  import java.io.FileNotFoundException;
4  import java.io.FileReader;
5  import java.io.IOException;
6
7  public class fileReaderClass {
8      String pname;
9      public fileReaderClass(String filename) {
10         pname=filename;
11     }
12
13     public String run() throws IOException {
14         String lsn= new String();
15         FileReader inputStream = null;
16         int x=0;
17         try {
18             inputStream = new FileReader(String.valueOf(pname));
19             int c;
20             while ((c = inputStream.read()) != -1) {
21                 lsn=lsn+String.valueOf((char)c);
22                 System.out.println(c);
23                 x++;
24             }
25         } catch (FileNotFoundException e) {
26             e.printStackTrace();
27         } catch (IOException e) {
28             e.printStackTrace();
29         } finally {
30             if (inputStream != null) {
31                 inputStream.close();
32             }
33         }
34         // System.out.println(x);
35         return lsn;
36     }
37 }
38
```

### isShiftdownConvert.java

```
1  package com.abdreshnayak.typingguru.JavaClasses;
2
3  public class isShiftdownConvert {
4      private String tempStr;
5      public String convert(String str){
6          tempStr=str;
7          switch (str){
8              case "`":
```

```
9         tempStr=~";
10         break;
11     case "1":
12         tempStr="!";
13         break;
14     case "2":
15         tempStr="@";
16         break;
17     case "3":
18         tempStr="#";
19         break;
20     case "4":
21         tempStr="$";
22         break;
23     case "5":
24         tempStr="%";
25         break;
26     case "6":
27         tempStr="^";
28         break;
29     case "7":
30         tempStr="&";
31         break;
32     case "8":
33         tempStr="*";
34         break;
35     case "9":
36         tempStr="(";
37         break;
38     case "0":
39         tempStr=")";
40         break;
41     case "-":
42         tempStr="_";
43         break;
44     case "=":
45         tempStr="+";
46         break;
47     case "q":
48         tempStr="Q";
49         break;
50     case "w":
51         tempStr="W";
52         break;
53     case "e":
54         tempStr="E";
55         break;
56     case "r":
57         tempStr="R";
58         break;
59     case "t":
60         tempStr="T";
61         break;
62     case "y":
63         tempStr="Y";
64         break;
65     case "u":
```



```
66         tempStr="U";
67         break;
68     case "i":
69         tempStr="I";
70         break;
71     case "o":
72         tempStr="O";
73         break;
74     case "p":
75         tempStr="P";
76         break;
77     case "[":
78         tempStr="{";
79         break;
80     case "]":
81         tempStr="}";
82         break;
83     case "\\":
84         tempStr="|";
85         break;
86     case "a":
87         tempStr="A";
88         break;
89     case "s":
90         tempStr="S";
91         break;
92     case "d":
93         tempStr="D";
94         break;
95     case "f":
96         tempStr="F";
97         break;
98     case "g":
99         tempStr="G";
100        break;
101    case "h":
102        tempStr="H";
103        break;
104    case "j":
105        tempStr="J";
106        break;
107    case "k":
108        tempStr="K";
109        break;
110    case "l":
111        tempStr="L";
112        break;
113    case ";":
114        tempStr=":";
115        break;
116    case "'":
117        tempStr="\'";
118        break;
119    case "z":
120        tempStr="Z";
121        break;
122    case "x":
```

```

123         tempStr="X";
124         break;
125     case "c":
126         tempStr="C";
127         break;
128     case "v":
129         tempStr="V";
130         break;
131     case "b":
132         tempStr="B";
133         break;
134     case "n":
135         tempStr="N";
136         break;
137     case "m":
138         tempStr="M";
139         break;
140     case ",":
141         tempStr="<";
142         break;
143     case ".":
144         tempStr=">";
145         break;
146     case "/":
147         tempStr="?";
148         break;
149     }
150     return tempStr;
151 }
152 }
153

```

#### langClass.java

```

1  package com.abdeshnayak.typingguru.JavaClasses;
2
3  import java.io.Serializable;
4  import java.util.HashMap;
5  import java.util.Map;
6
7  public class langClass implements Serializable {
8      public static final long serialVersionUID = 1L;
9      public Map<Integer,String> languages=new HashMap<>();
10 }
11

```

#### SorterClass.java

```

1  package com.abdeshnayak.typingguru.JavaClasses;
2
3
4  import java.util.*;
5
6  public class SorterClass {
7      public static boolean ASC = false;
8      public static boolean DESC = true;
9

```

```

10     public static Map<String, Integer> sortByComparator(Map<String, Integer>
unsortMap, final boolean order)
11     {
12
13         List<Map.Entry<String, Integer>> list = new LinkedList<Map.Entry<String,
Integer>>(unsortMap.entrySet());
14
15         // Sorting the list based on values
16         Collections.sort(list, (o1, o2) -> {
17             if (order)
18             {
19                 return o1.getValue().compareTo(o2.getValue());
20             }
21             else
22             {
23                 return o2.getValue().compareTo(o1.getValue());
24             }
25         });
26
27         // Maintaining insertion order with the help of LinkedList
28         Map<String, Integer> sortedMap = new LinkedHashMap<String, Integer>();
29         for (Map.Entry<String, Integer> entry : list)
30         {
31             sortedMap.put(entry.getKey(), entry.getValue());
32         }
33
34         return sortedMap;
35     }
36
37
38     public static void printMap(Map<String, Integer> map)
39     {
40         for (Map.Entry<String, Integer> entry : map.entrySet())
41         {
42             System.out.println("Key : " + entry.getKey() + " Value : " +
entry.getValue());
43         }
44     }
45 }
46

```

#### staticData.java

```

1     package com.abdeshnayak.typingguru.JavaClasses;
2
3     public class staticData {
4         public static String unvLanguage="English";
5         public static String username= new String();
6         public static int order=0;
7         public staticData(String str){
8             username=str;
9         }
10        public String getSrt(){
11            return username;
12        }

```

```

13     public static int timerSeconds;
14     public static boolean tempSwitch=false;
15     public static String myDir;
16     public static boolean keyBoardType=true;
17
18 }
19

```

### tempDatas.java

```

1  package com.abdheshnayak.typingguru.JavaClasses;
2
3  import com.abdheshnayak.typingguru.addUserClass;
4  import com.abdheshnayak.typingguru.keyboardController.htmlWriter;
5  import javafx.fxml.FXMLLoader;
6  import javafx.scene.control.ButtonType;
7  import javafx.scene.control.Dialog;
8  import javafx.scene.layout.GridPane;
9
10 import java.io.IOException;
11 import java.util.ArrayList;
12 import java.util.HashMap;
13 import java.util.Map;
14 import java.util.Optional;
15
16 public class tempDatas {
17
18
19     public Map<Integer,String> languages=new HashMap<>();
20     public isShiftdownConvert is = new isShiftdownConvert();
21     public ArrayList<String> lsnlist = new ArrayList<>();
22
23     public Thread t;
24     public Thread t1;
25     public Thread t2;
26
27     public boolean dialogptr=true;
28     public int spaceCounter;
29     public String toPrint;
30     public int cursorPointer=0;
31     public String lsn ;
32     public int lstcursorPointer=0;
33     public int filePointer =0;
34
35
36     public htmlWriter hw= new htmlWriter();
37     public toTypeController tw=new toTypeController();
38     public String[] temp1;
39     public String tempString;
40
41     public Map<Integer,String> lsnlistMap=new HashMap<>();
42
43
44     public addUserClass user = new addUserClass();
45
46

```

```

47
48     public void lsnNext() throws IOException {
49         filePointer++;
50         String filename;
51
52         filename=staticData.myDir+"/.src/lesson/"+lsnlist.get(filePointer);
53         fileReaderClass f = new fileReaderClass(filename);
54         lsn= f.run();
55     }
56
57
58     public void userDialog(GridPane MainGridPane) throws IOException,
59     ClassNotFoundException {
60         Dialog<ButtonType> dialog = new Dialog<>();
61         dialog.initOwner(MainGridPane.getScene().getWindow());
62         FXMLLoader fxmlLoader= new FXMLLoader();
63         fxmlLoader.setLocation(getClass().getResource("addUserWindow.fxml"));
64         try {
65             dialog.getDialogPane().setContent(fxmlLoader.load());
66         }catch (IOException e){
67             System.out.println("Exception:");
68             e.printStackTrace();
69             return;
70         }
71
72         dialog.getDialogPane().getButtonTypes().add(ButtonType.OK);
73         dialog.getDialogPane().getButtonTypes().add(ButtonType.CLOSE);
74         System.out.println("Ok");
75         Optional<ButtonType> result = dialog.showAndWait();
76         if(result.isPresent() && result.get() == ButtonType.OK) {
77             addUserClass newUser = fxmlLoader.getController();
78             if(newUser.save() == 1) {
79                 result=dialog.showAndWait();
80             }
81             System.out.println("OK Pressed");
82
83         }else {
84             System.out.println("Cancel Pressed");
85         }
86     }
87
88
89     public void addLessons() throws IOException {
90
91         //         languages.put(languages.size(),"English");
92         user.readLanguages();
93         //         languages.putAll(user.langclass.languages);
94         languages=user.langclass.languages;
95
96
97
98         //         languages.put(languages.size(),"English");
99         //         languages.put(languages.size(),"Hindi");
100        //         languages.put(languages.size(),"Nepali");

```

```
101 //      languages.put(languages.size(),"Punjabi");
102 //      languages.put(languages.size(),"Bangali");
103 //      languages.put(languages.size(),"Urdu");
104
105
106 //      lsnlist.add("Stories/STORY1.LSN");
107 //      MainWindowController.speedLabel.setText("hello");
108
109      lsnlist.add("TYPSE11.LSN");
110      lsnlist.add("TYPSE12.LSN");
111      lsnlist.add("TYPSE13.LSN");
112      lsnlist.add("TYPSE21.LSN");
113      lsnlist.add("TYPSE22.LSN");
114      lsnlist.add("TYPSE23.LSN");
115      lsnlist.add("TYPSE31.LSN");
116      lsnlist.add("TYPSE32.LSN");
117      lsnlist.add("TYPSE33.LSN");
118      lsnlist.add("TYPSE41.LSN");
119      lsnlist.add("TYPSE42.LSN");
120      lsnlist.add("TYPSE43.LSN");
121      lsnlist.add("TYPSE51.LSN");
122      lsnlist.add("TYPSE11.LSN");
123      lsnlist.add("TYPSE12.LSN");
124      lsnlist.add("TYPSE13.LSN");
125      lsnlist.add("TYPSE21.LSN");
126      lsnlist.add("TYPSE22.LSN");
127      lsnlist.add("TYPSE23.LSN");
128      lsnlist.add("TYPSE31.LSN");
129      lsnlist.add("TYPSE32.LSN");
130      lsnlist.add("TYPSE33.LSN");
131      lsnlist.add("TYPSE41.LSN");
132      lsnlist.add("TYPSE42.LSN");
133      lsnlist.add("TYPSE43.LSN");
134      lsnlist.add("TYPSE51.LSN");
135      lsnlist.add("TYPSE21.LSN");
136      lsnlist.add("TYPSE22.LSN");
137      lsnlist.add("TYPSE23.LSN");
138      lsnlist.add("TYPSE21.LSN");
139      lsnlist.add("TYPSE22.LSN");
140      lsnlist.add("TYPSE23.LSN");
141      lsnlist.add("TYPSE31.LSN");
142      lsnlist.add("TYPSE32.LSN");
143      lsnlist.add("TYPSE33.LSN");
144      lsnlist.add("TYPSE41.LSN");
145      lsnlist.add("TYPSE42.LSN");
146      lsnlist.add("TYPSE43.LSN");
147      lsnlist.add("TYPSE2N11.LSN");
148      lsnlist.add("TYPSE2N12.LSN");
149      lsnlist.add("TYPSE2N13.LSN");
150      lsnlist.add("TYPSE2N21.LSN");
151      lsnlist.add("TYPSE2N22.LSN");
152      lsnlist.add("TYPSE2N23.LSN");
153      lsnlist.add("TYPSE2N31.LSN");
154      lsnlist.add("TYPSE2N32.LSN");
155      lsnlist.add("TYPSE2N33.LSN");
156      lsnlist.add("TYPSE2N41.LSN");
157      lsnlist.add("TYPSE2N42.LSN");
```

```

158         lsnlist.add("TYP2N43.LSN");
159
160         lsnlistMap.put(lsnlistMap.size(), "Stories/STORY1.LSN");
161         lsnlistMap.put(lsnlistMap.size(), "Stories/STORY2.LSN");
162         lsnlistMap.put(lsnlistMap.size(), "Stories/STORY3.LSN");
163         lsnlistMap.put(lsnlistMap.size(), "Stories/STORY4.LSN");
164         lsnlistMap.put(lsnlistMap.size(), "Stories/STORY5.LSN");
165
166     }
167 }
168

```

## toTypeController.java

```

1  package com.abdheshnayak.typingguru.JavaClasses;
2
3  import java.io.FileNotFoundException;
4  import java.io.FileReader;
5  import java.io.FileWriter;
6  import java.io.IOException;
7  import java.util.ArrayList;
8  import java.util.HashMap;
9  import java.util.Map;
10
11  public class toTypeController implements Runnable{
12      private Map<String, Integer> capital = new HashMap<String,
Integer>();
13      private Map<String, Integer> All = new HashMap<String, Integer>();
14      private ArrayList<String> datas = new ArrayList<String>();
15      private class Datas{
16          String strFirst;
17          String strWord;
18          String strLast;
19      }
20      private Datas strings=new Datas();
21      private String
pname=staticData.myDir+"/.src/Images/keyboard/temptoTypehtmlfile.html";
22
23      public toTypeController() {
24
25
26          datas.add("<strong style=\"box-shadow:0px 0px 20px #14B524;
background-color:#86c086\">");
27          All.put("green", 0);
28          datas.add("<!DOCTYPE html PUBLIC \"-//W3C//DTD XHTML 1.0
Strict//EN\" \"https://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd\">\n" +
29              "<html xmlns=\"http://www.w3.org/1999/xhtml\"
xml:lang=\"en\" lang=\"en\">\n" +
30              "<head>\n" +
31              "    <meta http-equiv=\"Content-Type\"
content=\"text/html; charset=UTF-8\" />\n" +
32              "\n" +
33              "    <title>Online Keyboard</title>\n" +
34              "    <link rel=\"stylesheet\" type=\"text/css\"
href=\"styleoftoTypehtml.css\" />\n" +
35              "</head>\n" +

```

```

36         "<body>\n" +
37         "\n" +
38         "<div class=\"keyboard\" >\n" +
39         "    <p align=\"Center\"; style=\"Font-size:22px;\">";
40     All.put("htmlStart", All.size());
41     datas.add("</strong>");
42     All.put("greenClose", All.size());
43     datas.add("</p>\n" +
44             "</div>\n" +
45             "</body>\n" +
46             "</html>\n");
47     All.put("endHtml", All.size());
48
49 }
50
51 String lsn = new String();
52
53 public String read(String pname) throws IOException {
54     lsn = "";
55     FileReader inputStream = null;
56     int x = 0;
57     try {
58         inputStream = new FileReader(String.valueOf(pname));
59         int c;
60         while ((c = inputStream.read()) != -1) {
61             lsn = lsn + String.valueOf((char) c);
62         }
63     } catch (FileNotFoundException e) {
64         e.printStackTrace();
65     } catch (IOException e) {
66         e.printStackTrace();
67     } finally {
68         if (inputStream != null) {
69             inputStream.close();
70         }
71     }
72     //     System.out.println(lsn);
73     //     System.out.println("Read");
74     return lsn;
75 }
76
77 private Map<String,Integer> keys;
78
79
80 public void write(String strWord){
81     String tempst= "";
82
83     if(strWord.length()>400){
84         strWord=strWord.substring(strWord.length()-400);;
85     }
86
87
88     if(StaticData.unvLanguage.equals("English")){
89         tempst="";
90     }else {
91         tempst="font-size:30px;";
92     }

```



```

93         strings.strFirst=">"+strWord;
94         strings.strLast="</span>";
95         strings.strWord="";
96     }
97
98     public void write(String strFirst,String strSecond,String strThird)
throws IOException {
99
100         try {
101             String temp = "";
102             for (int i = 0; i < strFirst.length(); i++) {
103                 if (strFirst.charAt(i) == ' ') {
104                     temp = temp + " &nbsp;";
105                 } else {
106                     temp += strFirst.charAt(i);
107                 }
108             }
109             strings.strFirst = temp;
110             //      System.out.println(temp);
111
112             temp = "";
113             for (int i = 0; i < strSecond.length(); i++) {
114                 if (strSecond.charAt(i) == ' ') {
115                     temp = temp + " &nbsp;";
116                 } else {
117                     temp += strSecond.charAt(i);
118                 }
119             }
120             strings.strWord = temp;
121             //      System.out.println(temp);
122
123             temp = "";
124             for (int i = 0; i < strThird.length(); i++) {
125                 if (strThird.charAt(i) == ' ') {
126                     temp = temp + " &nbsp;";
127                 } else {
128                     temp += strThird.charAt(i);
129                 }
130             }
131             strings.strLast = temp;
132             //      System.out.println(temp);
133
134         }catch (Exception e){
135             System.out.println(e);
136         }
137     }
138
139
140     @Override
141     public void run() {
142         FileWriter outputStream = null;
143         lsn = "";
144         lsn = datas.get(All.get("htmlStart")) + strings.strFirst +
datas.get(All.get("green")) + strings.strWord +
datas.get(All.get("greenClose")) + strings.strLast +
datas.get(All.get("endHtml"));

```

```

145
146 //      System.out.println(lsn);
147     try {
148         outputStream = new FileWriter(String.valueOf(pname));
149         outputStream.write(lsn);
150     } catch (FileNotFoundException e) {
151         e.printStackTrace();
152     } catch (IOException e) {
153         e.printStackTrace();
154     } finally {
155         if (outputStream != null) {
156             try {
157                 outputStream.close();
158             } catch (IOException e) {
159                 e.printStackTrace();
160             }
161         }
162         try {
163             outputStream.close();
164         } catch (IOException e) {
165             e.printStackTrace();
166         }
167     }
168 //      System.out.println("Written");
169 }
170
171 }
172

```

### usrData.java

```

1  package com.abdreshnayak.typingguru.JavaClasses;
2
3  import java.io.Serializable;
4  import java.util.ArrayList;
5  import java.util.HashMap;
6  import java.util.Map;
7
8  public class usrData implements Serializable {
9      public static final long serialVersionUID = 1L;
10
11      public Map<String ,Integer> Score= new HashMap<>();
12      public Map<String ,String> users= new HashMap<>();
13
14      public ArrayList<String> userNames = new ArrayList<String>();
15
16      @Override
17      public String toString() {
18          return "usrData{" +
19              "Score=" + Score +
20              ", users=" + users +
21              '}';
22      }
23  }
24

```

