

Learner's Academy

A Back-End,
Administrative Portal.

Run Index.html!

1) Project Statement:

design and develop a backend administrative portal for the Learner's Academy. Learner's Academy is a school that has an online management system. The system keeps track of its classes, subjects, students, and teachers. It has a back-office application with a single administrator login.

2) Sprint Planning:

A. Product Backlog:

- i. **Plan Flow** of the Back End.
 - a. Algorithm for Login, Database Connectivity, Dynamic Page Generator.
 - b. Sorting Algorithms. Collections, Exception Handling.
 - c. Adding of Teachers, Students, Subjects from HTML interface (additional).
- ii. **Plan JAVA Classes, Servlets, .HTMLs and .JSPs.**
 - a. Algorithm for LoginController Servlet.
 - b. Algorithm for HomePage Servlet.
 - c. Algorithm for DatabaseAdder Servlet.
 - d. Algorithm for TimeScheduler Servlet.
 - e. Structures of Class, Teacher, Student, Subject classes.
- iii. **Create SQL file** to create schemas, tables and fill random values.
 - a. Create a '**Create Schema and Tables.sql**' file.
 - b. Create a '**Add Random Values.sql**' file.
- iv. **Create Classes, Servlets, .JSPs and Servlet** codes to display data from **Database**.
 - a. Create Classes, Teachers, Students, Subjects and Time Table classes.
 - b. Create 'HomePage.jsp', 'WelcomeDiv.html', 'Index.html', 'TimeTable.html' files. Configure 'web.xml' file.
 - c. Create LoginController Servlet.
 - d. Create HomePage Servlet.
 - e. Create DataBaseAdder Servlet.
 - f. Create TimeScheduler Servlet.
- v. **Test Run App.**
 - a. **Run** and **Debug** Application.

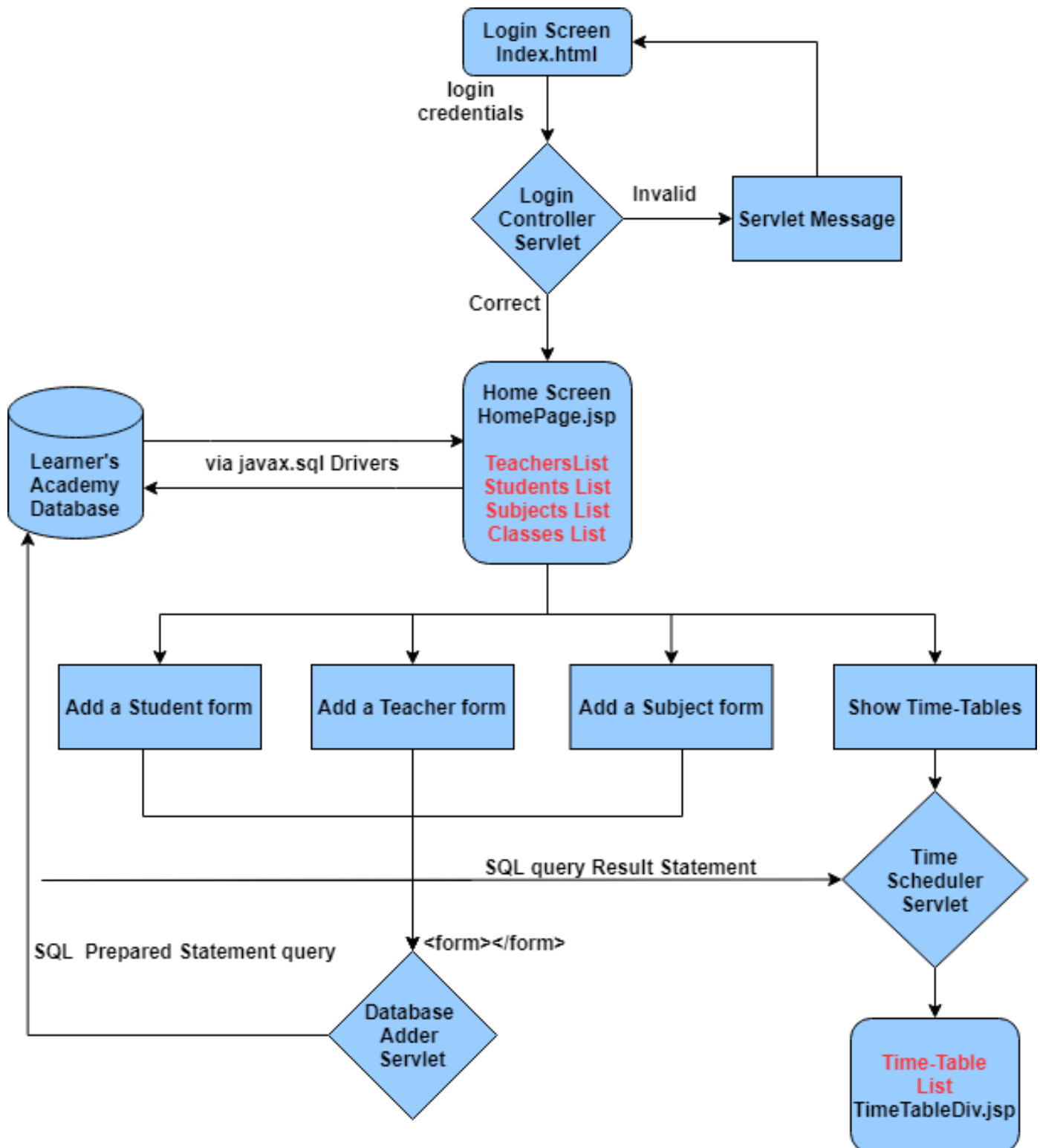
B. Sprint Table:

Sprint No.	Tasks	Estimation	Status
1.	Plan Flow of the Back End	4 hours	Pending
2.	Plan JAVA Classes, Servlets, .HTMLs and .JSPs	2 hours	Pending
3.	Create SQL file to create schemas, tables and fill random values	2 hours	Pending
4.	Create Classes, Servlets, JSPs and Servlet codes to display data from Database – PART I	8 hours	Pending
5.	Create Classes, Servlets, JSPs and Servlet codes to display data from Database – PART II	8 hours	Pending
6.	Test Run App	4 hours	Pending

C. Sprints:

i. Plan Flow of the Back End:

a. Algorithm for Login, Database Connectivity, Dynamic Page Generator:



b. Sorting Algorithms. Collections, Exception Handling:

In this we use ArrayLists<TimeTable> because using TimeTable[] timeTableArrays with Databases is difficult as getting the number of entries is harder with ResultSet rs as it is a Buffered writer type output. i.e. it is in different amounts of bytes rather than an explicit amount. We also use the comparator interface so that we can sort the ArrayList according to timeslot or dayOfTheWeek datamembers of the TimeTable class.

We use Exception Handlings such as SQLException, ArrayIndexOutOfBoundsException, IOException, ServletException, etc. These are important for reducing known crashes as is the use of Exception Handling. Also Errors are conveyed to user so that App can work as intended. Although we also use Types sparsely and correct TypeCasting of HTML sent requestParameters so that Exceptions can be avoided.

c. Adding of Teachers, Students, Subjects from HTML interface (additional):

We add some additional features to improve the quality of our product and save time as well as deliver more with this product. We have added HTML forms that send requests to a Servlet which then connects with database to insert additional entries. The features are:

- **Add a new Student.**
- **Add a new Teacher.**
- **Add a new Subject.**
- **Three different classes and their Time-Tables.**

ii. Plan JAVA Classes, Servlets, .HTMLs and .JSPs:

a. Algorithm for LoginController Servlet:

We use loginController servlet to check if the login credentials entered are the correct ones or invalid ones so that they can be redirected accordingly. If the credentials are wrong ones, we use RequestDispatcher to redirect the user back to index.html and include some other .html file or some generated text as feedback. If the credentials are right, we still use RequestDispatcher to forward the request to another servlet so that we can view the content that is usually sensitive and hidden to everyone but the user with correct login credentials.

b. Algorithm for HomePage Servlet:

We use HomePage servlet so that we can connect and communicate with the database and get the sensitive data stored in it for display for the user with correct login. Here we use RequestDispatcher and include all the different elements of pages which build the page that the user sees. Hence we use .include() method again here. We can alternatively use Java Bean and send all the objects to the user itself. But that would be rather tedious and contains the risk of data leaking if the login device was a public one.

The HomePage servlet uses Connection for connecting to database link, PrintWriter to generate dynamic response, ArrayList to store all the data got from the database, ResultSet as response from database then convert it to objects using constructors. It also uses ExecuteQuery to send commands to database, .sort() methods of ArrayList is used and Comparator is provided in the class itself wherever required. It also has the additional methods required so that the request and data the forms provide in the .html can be used to forwarded to another Servlet for breaking down, making objects for use and storing it in database in another servlet.

c. Algorithm for DatabaseAdder Servlet:

We use DatabaseAdder servlet to store additional objects such as Teachers, Students and Subjects. Data we get from the post method via the parameters is used to make Objects using the Teacher, Student or Subject constructors. These objects are then stored in memory as well as in database using the Connection, PreparedStatement objects and .execute() methods.

We use HTML hidden form field to locate the form number and hence the method that needs to be executed to be saved in the right table in the database. Form number also helps us change statements because every statement is different as every table is different and so are the incoming parameters. We also use HTML to fool proof the methods as the forms cannot be submitted without submitting all the required fields which are NOT NULL in the database and hence if the form is sent, the database is updated as user error has been removed. After submitting the form we use the RequestDispatcher, namely the .forward() method so that we can redirect the user to the same page but with the tables updated. We only show Teachers, Subjects, Classes and Students table on this page.

d. Algorithm for TimeScheduler Servlet:

We use this servlet to display all the Time-Tables of the three classes with the subjects and the assigned teachers we use generate most of this page dynamically. We use ResultStatement, Connection and .executeQuery() methods to get the information from database, convert them into objects so they remain in data pool, use constructors to turn them into objects i.e. TimeTable. Now we use for-each loops and PrintWriter to dynamically generate the web-page according to the amount of data.

e. Structures of Student, TimeTable, Subject, Teacher and Class:

We use create most of the data members as we want them to exist in the database but the Primary key which is identification is generated in the database using Auto-Increment and is given to the object hence the object does not have a setter method for the unique identification. We implement the Comparator method in TimeTable class as we need to sort it when we have it in a collection ArrayList.

iii. Create SQL files to create schema and fill random values:

a. Create a 'Create Schema and Tables.sql' file:

We create this file thinking that it will create a schema named 'learnersacademy' and also add tables students, teachers, subjects, classes and timetables. We create this file as fool proof so that if a schema already exists and/or the tables do too, the file is executed it first drops the schema as well as the tables then proceeds to create a new one along with the new tables. This file can also be used as reset if the tables are altered and/or functioning of app stops.

b. Create a 'Add Random Values.sql' file:

We create this file thinking that it will truncate the already existing values in the tables and Add the random values that we create so that the user can experience the app the way it was first given. Although beware the entered data will be lost if the file is executed. We create this file fool proof so that if the file is executed the tables are first truncated. This step is also necessary for proper functioning of the TimeScheduler servlet. This file can also be used as reset if the tables are altered and/or functioning of app stops.

iv. Create Classes, Servlets, JSPs and Servlet codes to display data from Database:

We create all the necessary java classes, java HttpServlets, JSP files and .HTML files and also debug the application where necessary to ensure proper working of the portal. The coding is done in Eclipse IDE using jdk 16 and JRE 1.8, the codes are uploaded to GitHub. GitHub link is at the start and end of document and footer of every page.

Sprint No.	Tasks	Estimation	Status
1.	Plan Flow of the Back End	4 hours	Done
2.	Plan JAVA Classes, Servlets, .HTMLs and .JSPs	2 hours	Done
3.	Create SQL file to create schemas, tables and fill random values	2 hours	Done
4.	Create Classes, Servlets, JSPs and Servlet codes to display data from Database -- PART I	8 hours	Done
5.	Create Classes, Servlets, JSPs and Servlet codes to display data from Database -- PART II	8 hours	Done
6.	Test Run App	4 hours	Done

- v. Test Run App:
 - c. Test and Debug the App:

The application was run and debugged several times. It was tweaked until all the grammatical errors and logical errors were sorted. A new feature was added where user can now add Teachers, Students and Subjects from the database itself.

3) Working of the App:

A. Logging in to the portal:

Login Screen:



Welcome to **Learners Academy** portal

(The Login details have already been entered because this is a prototype!)

Login

username
admin

password

login as admin!

Login Fail:



Welcome to **Learners Academy** portal

(The Login details have already been entered because this is a prototype!)

Invalid Credentials!

Login

username
admin

password

login as admin!

Home Page:

Welcome Admin!

Teachers
List

Teachers

Name	Date of Birth	Salary
Avery Parks	1991-12-17	30000.0
Wynne Bradley	1993-04-17	31500.0
Valentine Hunt	1985-08-17	29750.0

Students
List

Students

Name	Standard-Class
Jed Freeman	1 - A
Goodwin Walton	1 - A
Bennett Lindsey	1 - A
Winona Sherman	1 - B
Caldwell Moore	1 - B
Leonard Gordon	1 - B
Albert Dean	1 - C
Beatrice Kemp	1 - C
Tasha Marshman	1 - C

Subjects

Name
English
Mathematics
Science
Social Studies

Classes

Name
1 - A
1 - B
1 - C

Subjects
List

Classes
List

Student's Name:

Student's Standard:
☒ 1

Student's Class:
☐ A ☐ B ☐ C

Add a Student

Teacher's Name:

Teacher's Salary:

Date of Birth:
dd-mm-yyyy

Add a Teacher

Subject's Name

Add a Subject

Show Time-Table?

Add a Student form

Add a Teacher form

Add a Subject form

Show Time-Table Button

B. Master List of all Subjects for all Classes:

<u>Subjects</u>	<u>Classes</u>									
<table><tr><th>Name</th></tr><tr><td>English</td></tr><tr><td>Mathematics</td></tr><tr><td>Science</td></tr><tr><td>Social Studies</td></tr></table>	Name	English	Mathematics	Science	Social Studies	<table><tr><th>Name</th></tr><tr><td>1 - A</td></tr><tr><td>1 - B</td></tr><tr><td>1 - C</td></tr></table>	Name	1 - A	1 - B	1 - C
Name										
English										
Mathematics										
Science										
Social Studies										
Name										
1 - A										
1 - B										
1 - C										

C. Master List of all Teachers:

Teacher List

<u>Teachers</u>		
Name	Date of Birth	Salary
Avery Parks	1991-12-17	30000.0
Wynne Bradley	1993-04-17	31500.0
Valentine Hunt	1985-08-17	29750.0

Adding Teacher:

Teacher's Name:

Harsh Dedhia

Teacher's Salary:

31250

Date of Birth:

14-01-1993

Add a Teacher

Add a Teacher Form

Updated Portal List:

<u>Teachers</u>		
Name	Date of Birth	Salary
Avery Parks	1991-12-17	30000.0
Wynne Bradley	1993-04-17	31500.0
Valentine Hunt	1985-08-17	29750.0
Harsh Dedhia	1993-01-14	31250.0

Updated SQL List:

	idTeacher	name	DoB	salary
▶	1	Avery Parks	1991-12-17	30000
	2	Wynne Bradley	1993-04-17	31500
	3	Valentine Hunt	1985-08-17	29750
	4	Harsh Dedhia	1993-01-14	31250
✱	NULL	NULL	NULL	NULL

D. Master List of all classes:

<u>Classes</u>
Name
1 - A
1 - B
1 - C

E. Assigning Subjects:

Subjects List:

<u>Subjects</u>
Name
English
Mathematics
Science
Social Studies

Adding Subject:

Subject's Name

Add a Subject

Subject Form

Updated Portal List:

<u>Subjects</u>	
Name	
English	
Mathematics	
Quantum Mechanics	
Science	
Social Studies	

Updated SQL List:

	idSubjects	subjectName
▶	1	English
	2	Mathematics
	5	Quantum Mechanics
	3	Science
	4	Social Studies
*	NULL	NULL

F. Student Master List:

Students List:

<u>Students</u>	
Name	Standard-Class
Jed Freeman	1 - A
Goodwin Walton	1 - A
Bennett Lindsey	1 - A
Winona Sherman	1 - B
Caldwell Moore	1 - B
Leonard Gordon	1 - B
Albert Dean	1 - C
Beatrice Kemp	1 - C
Tasha Marshman	1 - C

Adding Student:

Student's Name:

Harsh Dedhia

Student's Standard:

☒ 1

Student's Class:

☐ A ☒ B ☐ C

Add a Student

Add a Student Form

Updated Portal List:

Students	
Name	Standard-Class
Jed Freeman	1 - A
Goodwin Walton	1 - A
Bennett Lindsey	1 - A
Winona Sherman	1 - B
Caldwell Moore	1 - B
Leonard Gordon	1 - B
Albert Dean	1 - C
Beatrice Kemp	1 - C
Tasha Marshman	1 - C
Harsh Dedhia	1 - B

Updated SQL List:

	idStudents	studentName	className	standard
▶	1	Jed Freeman	A	1
	2	Goodwin Walton	A	1
	3	Bennett Lindsey	A	1
	4	Winona Sherman	B	1
	5	Caldwell Moore	B	1
	6	Leonard Gordon	B	1
	7	Albert Dean	C	1
	8	Beatrice Kemp	C	1
	9	Tasha Marshman	C	1
	10	Harsh Dedhia	B	1
*	NULL	NULL	NULL	NULL

4) Git and GitHub:

Now we link git on PC and git repository on git hub using git bash. Then we use git commit and git push commands to upload to repository and give a commit message.

```
HD@DESKTOP-JMBP1R8 MINGW64 ~/Desktop/proj/SL/LearnersAcademy
$ rd .git /S/Q
bash: rd: command not found

HD@DESKTOP-JMBP1R8 MINGW64 ~/Desktop/proj/SL/LearnersAcademy
$ git init
Initialized empty Git repository in C:/Users/HD/Desktop/proj/SL/LearnersAcademy/.git/

HD@DESKTOP-JMBP1R8 MINGW64 ~/Desktop/proj/SL/LearnersAcademy (master)
$ git add .
warning: LF will be replaced by CRLF in Add Random Values SQLFile.txt.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in create schema and tables SQLFile.txt.
The file will have its original line endings in your working directory

HD@DESKTOP-JMBP1R8 MINGW64 ~/Desktop/proj/SL/LearnersAcademy (master)
$ git commit -m "First iteration commit"
[master (root-commit) c1de7ab] First iteration commit
71 files changed, 1843 insertions(+)
create mode 100644 Add Random Values SQLFile.txt
create mode 100644 Classes.java
create mode 100644 DataBaseAdder.java
create mode 100644 HomePage.java
create mode 100644 HomePage.jsp
create mode 100644 LNAcad/.classpath
create mode 100644 LNAcad/.project
create mode 100644 LNAcad/.settings/.jsdt.scope
create mode 100644 LNAcad/.settings/org.eclipse.jdt.core.prefs
create mode 100644 LNAcad/.settings/org.eclipse.wst.common.component
create mode 100644 LNAcad/.settings/org.eclipse.wst.common.project.facet.core.xml
create mode 100644 LNAcad/.settings/org.eclipse.wst.jsdt.ui.superType.container
create mode 100644 LNAcad/.settings/org.eclipse.wst.jsdt.ui.superType.name
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/Classes.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/DatabaseAdder.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/HomePage.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/LoginController.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/Student.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/Subject.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/Teacher.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/TimeScheduler.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/TimeTable.class
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/Classes.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/DatabaseAdder.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/HomePage.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/LoginController.java
```

```

create mode 100644 LNAcad/build/classes/com/SL/LNAcad/HomePage.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/LoginController.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/Student.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/Subject.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/Teacher.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/TimeScheduler.class
create mode 100644 LNAcad/build/classes/com/SL/LNAcad/TimeTable.class
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/Classes.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/DatabaseAdder.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/HomePage.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/LoginController.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/Student.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/Subject.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/Teacher.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/TimeScheduler.java
create mode 100644 LNAcad/src/main/java/com/SL/LNAcad/TimeTable.java
create mode 100644 LNAcad/src/main/webapp/HomePage.jsp
create mode 100644 LNAcad/src/main/webapp/META-INF/MANIFEST.MF
create mode 100644 LNAcad/src/main/webapp/TimeTableDiv.html
create mode 100644 LNAcad/src/main/webapp/WEB-INF/lib/mysql-connector-java-8.0.
26.jar
create mode 100644 LNAcad/src/main/webapp/WEB-INF/web.xml
create mode 100644 LNAcad/src/main/webapp/WelcomeDiv.html
create mode 100644 LNAcad/src/main/webapp/index.html
create mode 100644 Learners Academy.docx
create mode 100644 LoginController.java
create mode 100644 Student.java
create mode 100644 Subject.java
create mode 100644 Teacher.java
create mode 100644 TimeScheduler.java
create mode 100644 TimeTable.java
create mode 100644 TimeTableDiv.html
create mode 100644 WelcomeDiv.html
create mode 100644 create schema and tables SQLFile.txt
create mode 100644 img/AddStudent_part1_form.JPG
create mode 100644 img/AddStudent_part2_SQLTeacherList.JPG
create mode 100644 img/AddStudent_part2_UpdatedTeacherList.JPG
create mode 100644 img/AddSubject_part1_form.JPG
create mode 100644 img/AddSubject_part2_SQLTeacherList.JPG
create mode 100644 img/AddSubject_part2_UpdatedTeacherList.JPG
create mode 100644 img/AddTeacher_part1_form.JPG
create mode 100644 img/AddTeacher_part2_SQLTeacherList.JPG
create mode 100644 img/AddTeacher_part2_UpdatedTeacherList.JPG
create mode 100644 img/ClassesForSubjects.JPG
create mode 100644 img/ClassesList.JPG
create mode 100644 img/Learners Academy Algorithm Flow.png
create mode 100644 img/LoggingIn_part1.JPG
create mode 100644 img/LoggingIn_part2.JPG
create mode 100644 img/LoginFail.JPG

```

```























create mode 100644 img/LoginFail.JPG
create mode 100644 img/Master List of all Subjects for all Classes.JPG
create mode 100644 img/StudentsList.JPG
create mode 100644 img/SubjectsList.JPG
create mode 100644 img/TeachersList.JPG
create mode 100644 index.html
create mode 100644 web.xml
create mode 100644 ~$arners Academy.docx
create mode 100644 ~WRL1759.tmp

HD@DESKTOP-JMBP1R8 MINGW64 ~/Desktop/proj/SL/LearnersAcademy (master)
$ git remote add origin https://github.com/dedhiah10/Learner-s-Academy.git

HD@DESKTOP-JMBP1R8 MINGW64 ~/Desktop/proj/SL/LearnersAcademy (master)
Writing objects: 100% (77/77), 2.92 MiB | 5.44 MiB/s, done.
Total 77 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), done.
To https://github.com/dedhiah10/Learner-s-Academy.git
* [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

```

GitHub Repository: <https://github.com/dedhiah10/Learner-s-Academy>

	dedhiah10 First iteration commit	c1de7ab 10 minutes ago	 1 commit
	LNAcad	First iteration commit	10 minutes ago
	img	First iteration commit	10 minutes ago
	Add Random Values SQLFile.txt	First iteration commit	10 minutes ago
	Classes.java	First iteration commit	10 minutes ago
	DataBaseAdder.java	First iteration commit	10 minutes ago
	HomePage.java	First iteration commit	10 minutes ago
	HomePage.jsp	First iteration commit	10 minutes ago
	Learners Academy.docx	First iteration commit	10 minutes ago
	LoginController.java	First iteration commit	10 minutes ago
	Student.java	First iteration commit	10 minutes ago
	Subject.java	First iteration commit	10 minutes ago
	Teacher.java	First iteration commit	10 minutes ago
	TimeScheduler.java	First iteration commit	10 minutes ago
	TimeTable.java	First iteration commit	10 minutes ago
	TimeTableDiv.html	First iteration commit	10 minutes ago
	WelcomeDiv.html	First iteration commit	10 minutes ago
	create schema and tables SQLFile.txt	First iteration commit	10 minutes ago
	index.html	First iteration commit	10 minutes ago
	web.xml	First iteration commit	10 minutes ago
	~\$amers Academy.docx	First iteration commit	10 minutes ago
	~WRL1759.tmp	First iteration commit	10 minutes ago

GitHub Repository: <https://github.com/dedhiah10/Learner-s-Academy>