**Chapter 1**

**INTRODUCTION**

Musyqa is an online music learning portal for popular western classical music.

**1.1 Western Classical Music:-**

The key characteristic of Western Classical Music, that distinguishes it from other classical music is that the reportoire tends to be written down in **musical notation,** creating a musical part of a score. This score determines details of rhythm, pitch, where two or more mucisians are involved, how various parts are coordinated.

The written quality of the music has enabled a high level of complexity within them: [J.S. Bach](https://en.wikipedia.org/wiki/J.S._Bach)'s [fugues](https://en.wikipedia.org/wiki/Fugue), for instance, achieve a remarkable marriage of boldly distinctive melodic lines weaving in [**counterpoint**](https://en.wikipedia.org/wiki/Counterpoint) yet creating a coherent [**harmonic logic**](https://en.wikipedia.org/wiki/Diatonic_function) that would be impossible in the heat of live improvisation. The use of written notation also preserves a record of the works and enables Classical musicians to perform music from many centuries ago. Musical notation enables 2000s-era performers to sing a choral work from the 1300s Renaissance era or a 1700s Baroque concerto with many of the features of the music (the melodies, lyrics, forms, and rhythms) being reproduced.

**1.2 Features of Musyqa**

The website serves the user by facilitating him/her with two types of courses , namely,

**i)** **Basic Music Theory:-**

One of the biggest advantages of western classical music is the availability of the staff notations. Which are archived and preserved. It is easier for students to learn western music once they learn the basics of music theory and reading sheet music. Our website provides the training to read/write music notations on the staff and also give an insight about the music theory.

**ii) Grade-wise Listing:-**

Lists the popular genres. The user gets to choose a particular genre. Like composer course, genre course also has two options. One to learn the popular music as was maintained in the composer course or select a particular composer who has specialised in that genre and learn all his music in increasing order of difficulty.

The user gets the following facities while learning the music:

* Description of the music
* Audio file
  + in mp3 format
  + downloadable
* Sheet Music
* in pdf format
* downloadable
* Embedded youtube video

**1.3 Summary**

Since each and every musical piece of western classic has a recorded musical score, it plays a huge role for students and musicians alike to learn and improvise on their skills of music. This website tends to reduce the burden on western classical music students by compiling all the music courses gradewise and provides a holistic environment for them to learn.

**Chapter 2**

**SOFTWARE ENVIRONMENT**

**2.1 RESTful API**

**Representational state transfer** (**REST**) or **RESTful** [Web services](https://en.wikipedia.org/wiki/Web_service) are one way of providing interoperability between computer systems on the [Internet](https://en.wikipedia.org/wiki/Internet). REST-compliant Web services allow requesting systems to access and manipulate textual representations of [Web resources](https://en.wikipedia.org/wiki/Web_resource) using a uniform and predefined set of [stateless](https://en.wikipedia.org/wiki/Stateless_protocol) operations.

In a RESTful Web service, requests made to a resource's [URI](https://en.wikipedia.org/wiki/URI) will elicit a response that may be in [XML](https://en.wikipedia.org/wiki/XML), [HTML](https://en.wikipedia.org/wiki/HTML), [JSON](https://en.wikipedia.org/wiki/JSON) or some other defined format. The response may confirm that some alteration has been made to the stored resource, and it may provide [hypertext](https://en.wikipedia.org/wiki/Hypertext) links to other related resources or collections of resources. Using [HTTP](https://en.wikipedia.org/wiki/HTTP), as is most common, the kind of operations available include those predefined by the [HTTP verbs](https://en.wikipedia.org/wiki/HTTP_verbs) GET, POST, PUT, DELETE and so on.

The architectural properties affected by the constraints of the REST architectural style are:

* Performance
* Scalability
* Simplicity of uniform interface
* Modifiability of components to meet changing needs
* Visibility of communication between components by service agents
* Portability of components by moving program code with the data

**2.2 Database Management Systems (DBMS)**

DBMS is a collection of programs that enables users to create and maintain a database. The DBMS is a general purpose software system that facilitates the processes of defining, constructing, manipulating and sharing databases among various users and applications.

A  **Relational database** is a  database that has a collection of tables of data items, all of which is formally described and organized according to the relational model. Data in a single

table represents a relation, from which the name of the database type comes. In typical solutions, tables may have additionally defined relationships with each other. In the relational model, each table schema must identify a column or group of columns, called the *primary key*, to uniquely identify each row. A relationship can then be established between each row in the table and a row in another table by creating a *foreign key*, a column or group of columns in one table that points to the primary key of another table.

**Structured Query Language (SQL)**

The ANSI standard SQL provides basic functions for data manipulation, transaction control, and record retrieval from the database. However, most end users interact with Oracle through application that provides an interface that hides the underlying SQL and its complexity.

SQL uses the terms table, row, and column for relation, tuple, and attribute, respectively. The SQL commands for data definition are CREATE, ALTER and DROP.

* CREATE – this command is used to define the structure of the tables.
* ALTER – this command is used to change the structure of the tables.
* DROP – this command can be used to delete the table structure.

**Statements in SQL**

Following are the important statements used in SQL.

1. SELECT - Used to retrieve the information from the relation.
2. INSERT - Used to insert the new values to the relation.
3. DELETE - used to delete one or more existing tuples from the relation.
4. UPDATE - Used to update already existing values in the relation.

**Aggregate Functions in SQL**

Following aggregate functions are provided by the SQL.

(i) COUNT - Returns number of tuples.

(ii) SUM - Returns sum of entries in a column.

(iii) MAX - Returns Maximum value from an entire column.

(iv) MIN - Returns Minimum value from an entire column.

(v) AVG - Returns Average of all the entries in a column.

**Constraints in SQL**

Following constraints are provided by the SQL.

1. NOT NULL - Column should contain some value.
2. PRIMARY KEY - Should not allow duplicate and null values to a column.

(iii) UNIQUE - Each value of a column should be unique.

**Chapter 3**

**SYSTEM SPECIFICATION**

1. **Hardware Requirements**

* System : Intel® Core™ i3-4005U
* Hard Disk : 500 GB.
* Ram : 4 GB.

1. **Software Requirements**

* Operating system : Ubuntu 16.04.
* Front end : Angular2, Bootstrap, HTML, Typescript
* Back end : Express.js, MySQL 5.7.5
* Server : Node.js server
* Platform : MEAN (Mysql Express Angular Node)

1. **Overview of Tools/Software**

* **Node.js**

**Node.js** is an [open-source](https://en.wikipedia.org/wiki/Open-source_software), [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [JavaScript](https://en.wikipedia.org/wiki/JavaScript) [run-time environment](https://en.wikipedia.org/wiki/Runtime_system) for executing JavaScript code [server-side](https://en.wikipedia.org/wiki/Server-side). Historically, JavaScript was used primarily for [client-side scripting](https://en.wikipedia.org/wiki/Client-side_scripting), in which scripts written in JavaScript are embedded in a webpage's HTML, to be run client-side by a JavaScript engine in the user's web browser. Node.js enables JavaScript to be used for [server-side scripting](https://en.wikipedia.org/wiki/Server-side_scripting), and runs scripts server-side to produce [dynamic web page](https://en.wikipedia.org/wiki/Dynamic_web_page) content *before* the page is sent to the user's web browser. Consequently, Node.js has become one of the foundational elements of the "JavaScript everywhere" paradigm.

* **Express.js**

Express.js, or simply Express, is a [web application framework](https://en.wikipedia.org/wiki/Web_application_framework) for [Node.js](https://en.wikipedia.org/wiki/Node.js), released as [free and open-source software](https://en.wikipedia.org/wiki/Free_and_open-source_software) under the [MIT License](https://en.wikipedia.org/wiki/MIT_License). It is designed for building [web applications](https://en.wikipedia.org/wiki/Web_application) and [APIs](https://en.wikipedia.org/wiki/API). It is the [*de facto*](https://en.wikipedia.org/wiki/De_facto) standard server framework for Node.js.

* **Angular2**

Angular2 is the second version of AngularJS (commonly referred to as "Angular.js") is a JavaScript-based [open-source](https://en.wikipedia.org/wiki/Open-source_software) front-end [web application framework](https://en.wikipedia.org/wiki/Web_application_framework) mainly maintained by [Google](https://en.wikipedia.org/wiki/Google) and by a community of individuals and corporations to address many of the challenges encountered in developing [single-page applications](https://en.wikipedia.org/wiki/Single-page_application).

* **HTML**

HTML is the main mark-up language for displaying web pages and other information that can be displayed in a web browser. HTML is written inthe form of HTML elements consisting of tags enclosed in angle brackets (like<html>), within the web page content.

* **MySQL**

MySQL officially, but also called “MY Sequel” is the world’s most used open source relational database management system (RDBMS) that runs as server providing multi-user access to a number of databases.The SQL phrase stands for Structure Query Language.

**Chapter 4**

**SYSTEM DESIGN**

**4.1 ER Diagram**

An Entity-Relationship (ER) model is an abstract way to describe a database. It is a popular high-level conceptual data model. *Entity relationship diagrams* (ER diagrams) are used to present the diagrammatic notations associated with ER model (ref fig:4.1).

**4.1.1 Notations for ER Diagram**

***Symbols Meaning***

Entity

Weak Entity

Relationship

Identifying Relationship

Attribute

Multi-valued Attribute

Key Attribute

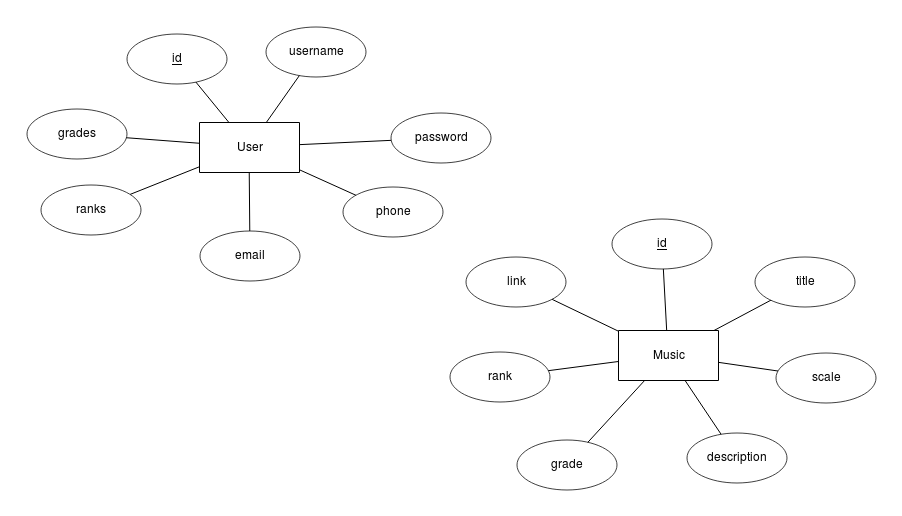
Composite Attribute

1 N Total Participation of E2

Cardinality Ratio 1: N for E1: E2

**fig : 4.1**

**4.1.2 ER Diagram for Musyqa**

**fig : 4.2**

The Musyqa Database consists of two entity types, viz. Music, and User. (ref fig: 4.2)

The Music entity type contains the details of a musical piece.

The User entity type contains the details of a user.

**4.2 Schema Diagram**

The description of database is called the *Database Schema*, which is specified during database design and is not expected to change frequently. Most data models have certain conventions for displaying schemas as diagrams. A displayed schema is called a schema diagram. The schema diagram for Musyqa (ref fig: 4.3) is given below

USER

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **id** | username | password | email | phone | grades | ranks |

MUSIC

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **id** | Title | Description | Scale | Link | Grade | rank |

**Fig 4.3** Schema diagram for Musyqa

**Chapter 5**

**IMPLEMENTATION**

**5.1 Database Tables/Relations**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Slno** | **Table** | |  |  |  | | --- | --- | --- | | **Key** | **Attribute** | **Type** | |
| 1. | User | |  |  |  | | --- | --- | --- | | Primary key | id | int | |  | username | varchar(255) | |  | password | varchar(255) | |  | email | varchar(255) | |  | Phone | varchar(255) | |  | grades | int | |  | ranks | int | |
| 2. | Music | |  |  |  | | --- | --- | --- | | Primary key | id | int | |  | title | varchar(255) | |  | description | text | |  | scale | varchar(255) | |  | grade | int | |  | rank | int | |  | link | varchar(255) | |

**Table 5.1** Structure of tables**Tables:-**

The following is a description ( ref table 5.1 ) of each table

**1.** **User** : Details of a user.

id : Primary key

username: username for a user to use in login sessions.

password : password for the user

email : email if of the user for communication.

phone: phone number for communication.

Grades: number of grades completed/under completion by the user

Ranks: number of music pieces completed/under completion by the user

**2. Music :** Details of music

id : Primary Key

title : title of the music

description : A few lines of description for the music

scale : musical scale of the piece

grade : grade of the music

rank: rank of the music in the given grade

link : embedded youtube link

**5.2 Node Scripts**

Below are the few Node scripts used in the website:

* ***db\_sequelize.js :*** defines a sequelize object to connect to the database using the information such as *host, username, password and database name* from the config fileand returns the connection object.
* ***server.js :*** main node script of Musyqa which runs the server creating necessary connections to the controllers, routes, synchronising the database and then listening at the specified port.
* ***routes.js :*** maintains routes for all the open rest points.
* ***controllers.js:*** the file which stores all the controllers assembled to be exported for further use.
* ***getName.js :*** used to retrieve username of user after he/she logs in.
* ***getCompletedRanks.js*** *:* get number of ranks completed by the user in the last grade..
* ***getCompletedRaks.js***: get completed grades by the user.
* ***downloader.js :*** used to download audio file, sheet music and certificate for the user..
* ***getMusic.js :*** displays the details of a music by its grade and rank.
* ***updateRank.js:*** updates the grade and rank of the user once he/she completes a music piece.

**5.3. Angular Components and Services**

The data are sent in a JSON format from the REST API to the lite-server of Angular.

Below are the few angular components used at the front end

* ***home.component.ts***is used to display the home page.
* ***about.component.ts***is used to display about the music site
* ***signup.component.ts*** is used to register a user.
* ***login.component.ts*** is used for a user to login.
* ***grade.component.ts***is used to display list of grades.
* ***list.component.ts*** is used to display list of music of a given grade.
* ***logout.component.ts*** is used to display the user that he/she has logged out.
* ***profile.component.ts*** is used to greet a user as a Maestro once he/she logs in.
* ***app.routing.ts*** contains the routes for respective components when a url is opened.
* ***app.module.ts***is the master component which imports all individual components and acts as a master module.
* ***posts.service.ts*** is used to define various functions to send a http GET/POST request to the REST API server.

**Chapter 6**

**CONCLUSION**

This project “Musyqa : The contemporary portal for learning classical music” provides graphical user friendly interface to learn various popular western classical music which are ranked by their grades and ranks.

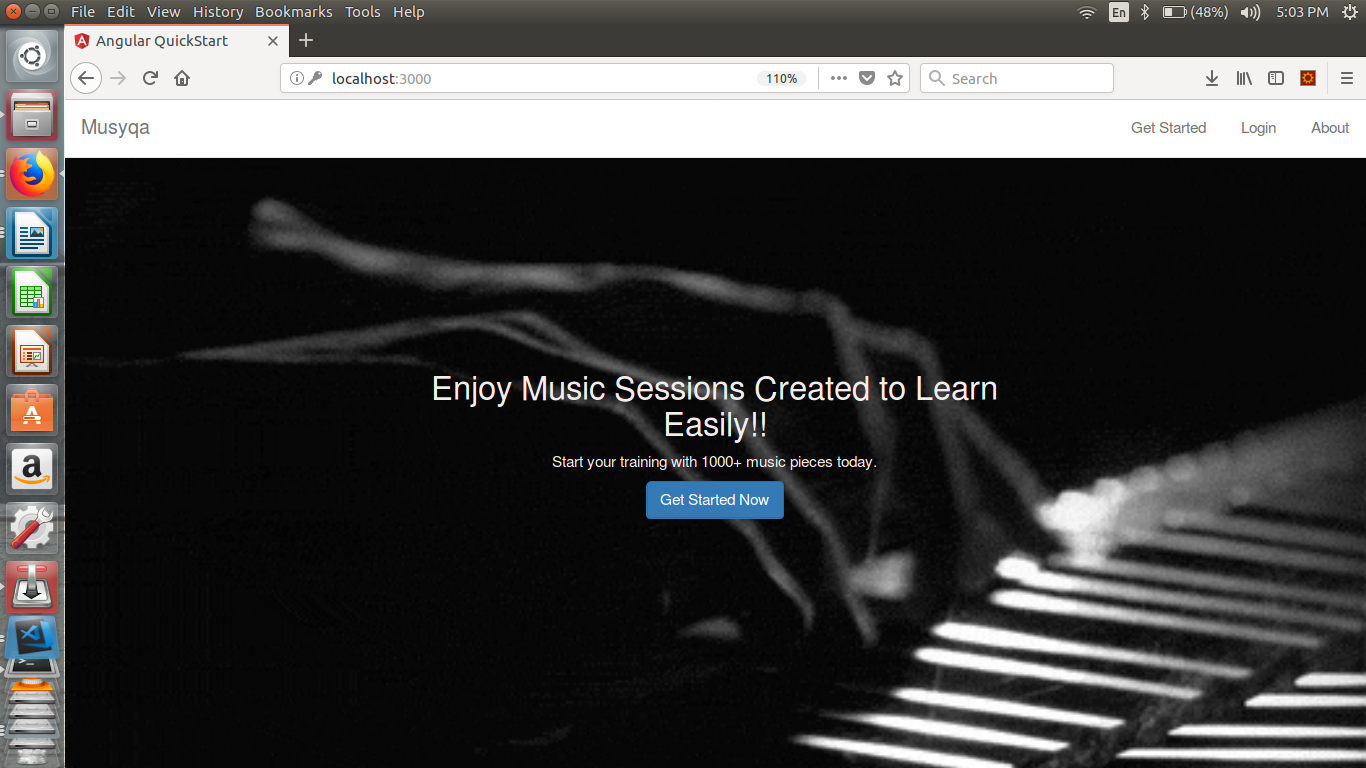
Growing popularity of online learning courses has created the need for having this portal for western classical music for free and of fine quality.

Future enhancement of Musyqa would be making it more interactive with online live video tutorials.

This project will definitely be of utmost use for those who want to learn popular classical music. Although the sheet musics are available, students have a tough time to decide what to play, from where to start, where to get an audio file, etc. There isnt a single portal where all the audio, sheet music along with the grades being compiled in a intuituve way so that it becomes easy for the student. Using this portal, user can easily decide what to play, along with facilities of sheet music , mp3 audio and a embedded youtube link.

Digitalisation is a global issue. Every application is now being developed on a computerized platform. It is a small effort by us to compile all the required equipments and store in a database to learn a music. The project has been a great experience. It has exposed me to the interesting world of Web development languages. It has inspired me to create better applications using Web Development languages.

**SCREENSHOTS**

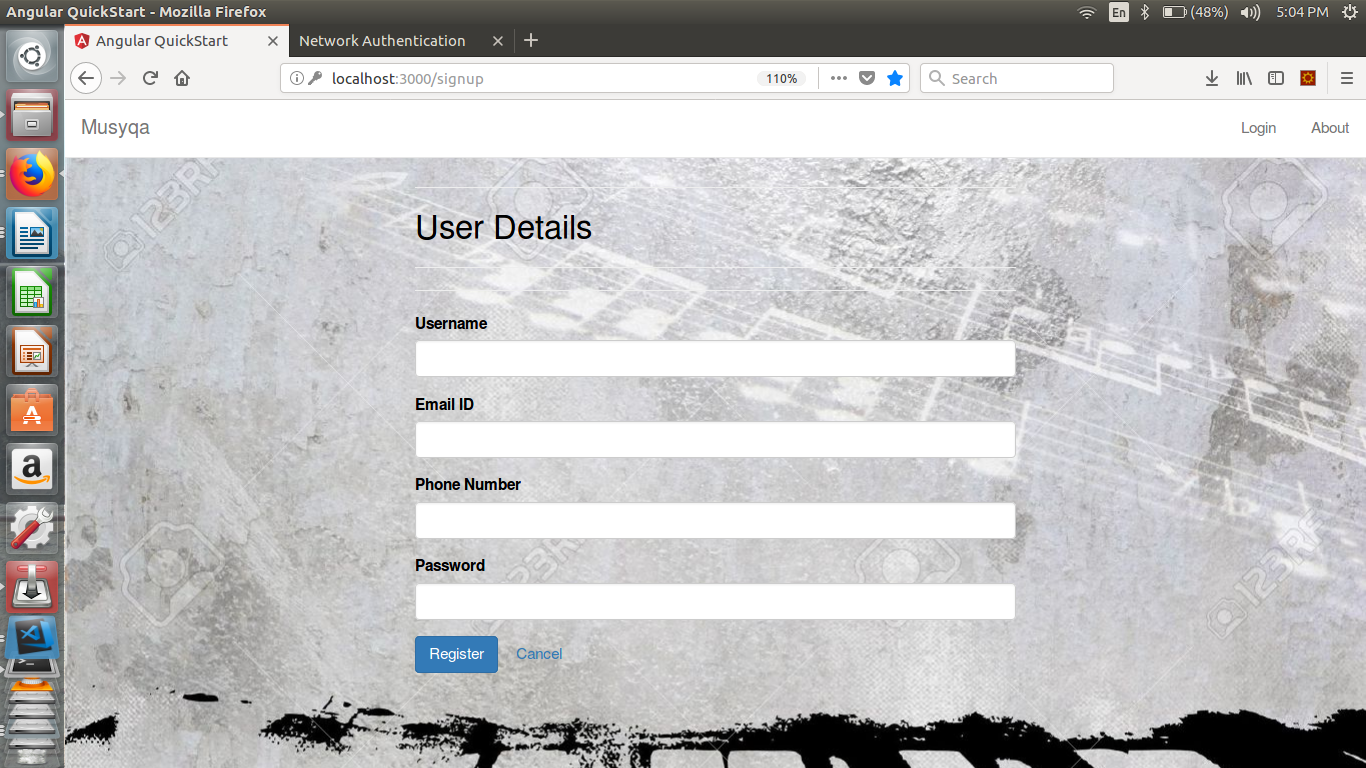
****

**fig 6.1**

The home page of musyqa (ref fig 6.1)

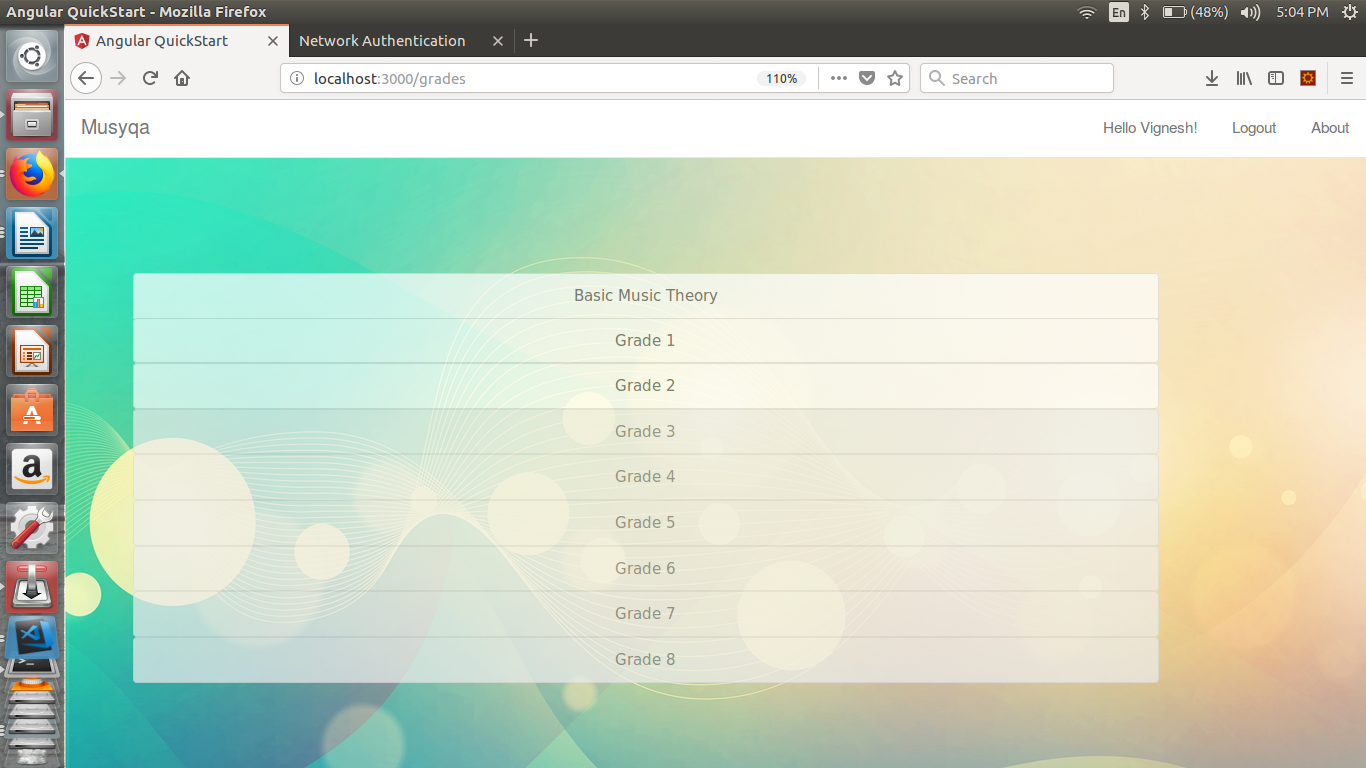
It contains links to about page and

login page and registeration page



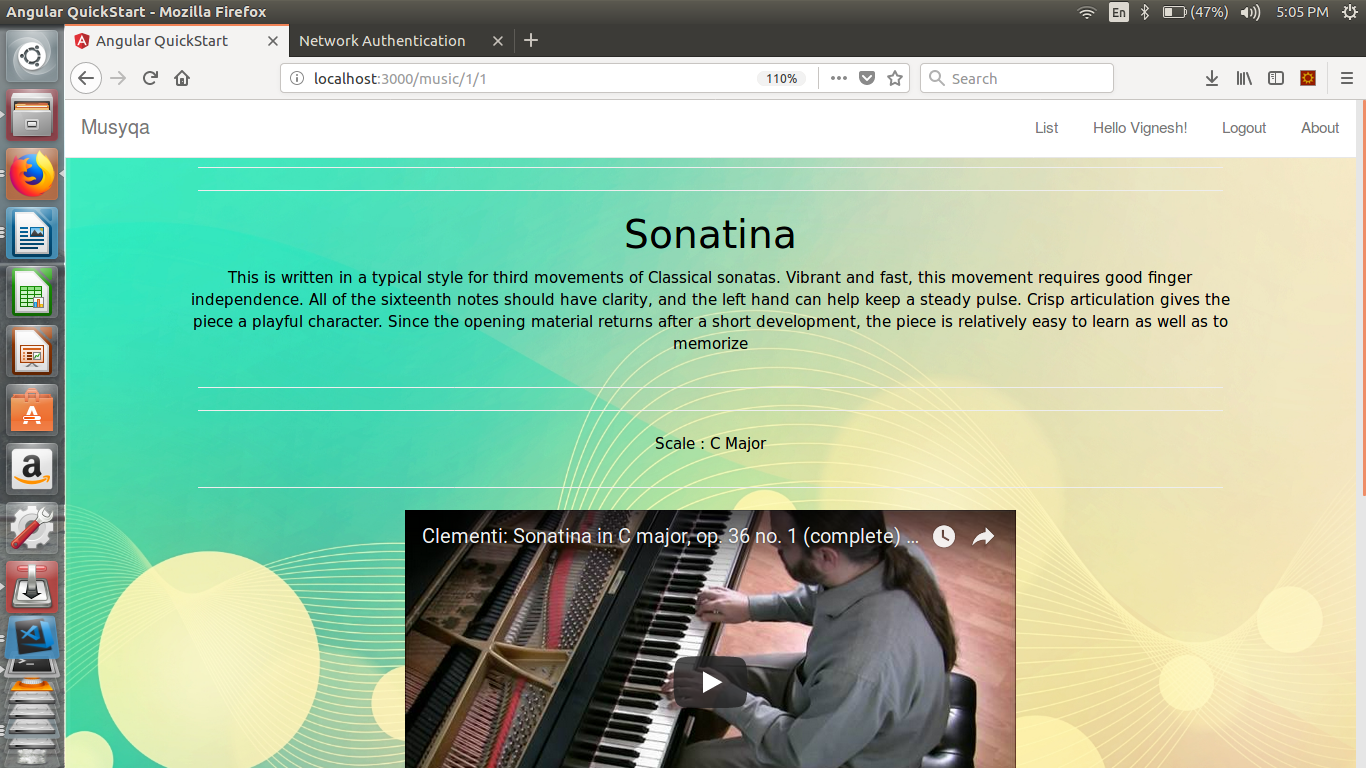
**fig 6.2**

Registration page(ref fig 6.2)

**3.**

**fig 6.3**

list of grades(ref fig 6.3)

****

**fig 6.4**

A music page (ref fig 6.4)

**BIBLIOGRAPHY**

1. Elmasri and Navathe, *Fundamentals of Database Systems*, Third Edition, Addison-Wesley, 2000.
2. Hege Refsnes, *Learn HTML and CSS with w3schools*, First Edition, Wiley Publishing, 2010.
3. [http://www.w3schools.com/php/](http://www.w3schools.com/)
4. [http://www.php.net](http://www.php.net/)
5. [http://www.html.net](http://www.html.net/)
6. [http://www.stackoverflow.com](http://www.stackoverflow.com/)
7. <https://angular.io/>
8. <https://nodejs.org/>
9. <https://docs.sequelizejs.com/>

[10]Robert Sebasta, Programming the World Wide Web, Published August 1st 2007 by Addison Wesley Longman