ASSIGNMENT 3 – GROUP PROJECT

DOCUMENTATION

Full-stack application development

Team Members:

Meghana Adithi Bongu - G01446886

Tarun Naga Sai Chadaram - G01445928

Bhavishya kollipara - G01455074

Deepika Hemant Tendulkar - G01451861

1. **Objective:** For this survey application project, it depends upon 2 main subcomponents in it which are filling the survey form and representing survey forms given by the user. The ability to delete and edit from the list of survey forms.

2. **Technology Stack**

- Front end : Angular (Typescript, HTML, CSS)are used for the Survey-form for managing the html ,css and logic behind it using type script file.
- Backend: Spring Boot (Java, Hibernate ORM) are used for communication between the frontend (angular) and the database.
- Database: MySQL(here is where everything related to data is stored and CRUD operations are rawly performed.We MySQL workbench GUI to have broader understanding of how the data is carried on the both hands.

Features of Survey application:

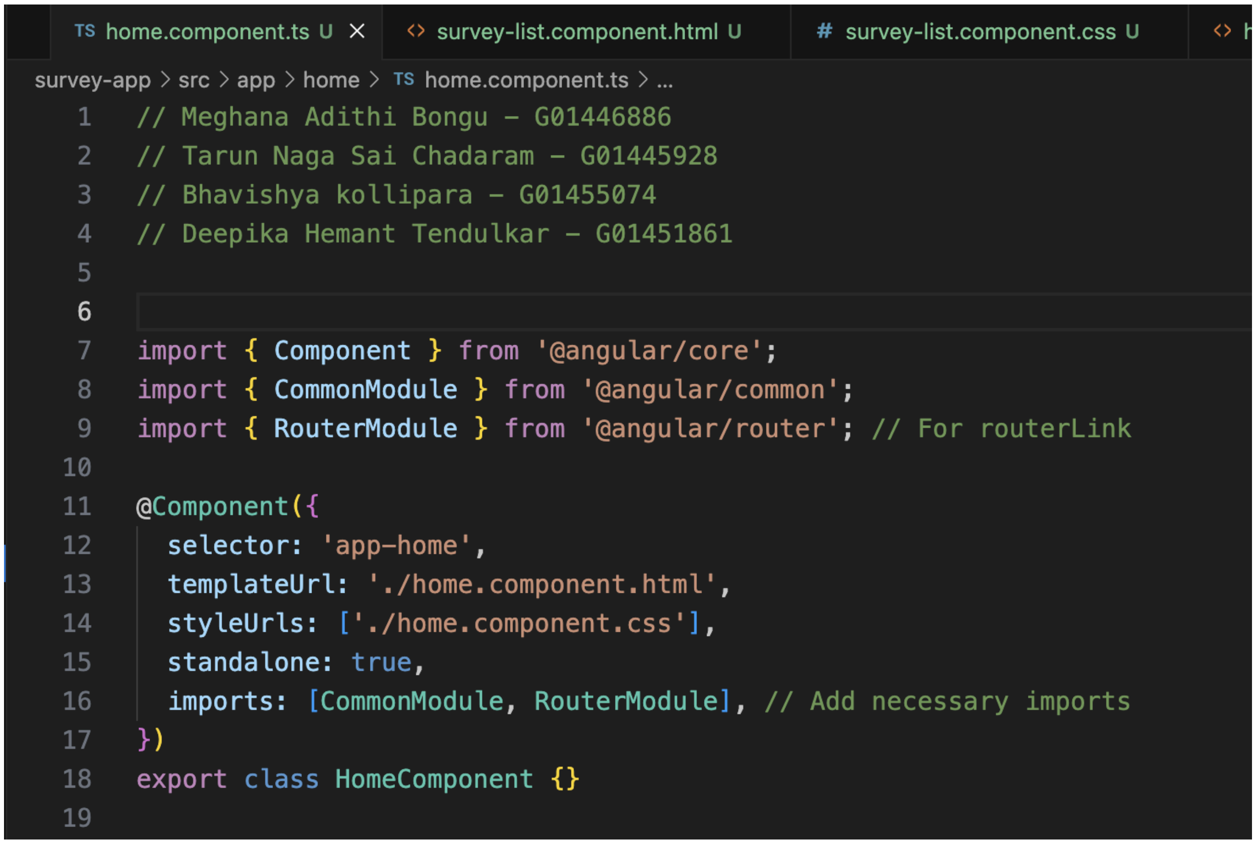
- **Home page:** This would be the landing paging of the application.Where it consists of buttons for redirecting to survey form page and survey list page.
- **Survey Form:** Users can fill out detailed surveys, including personal information and preferences.

- **Survey List:** A comprehensive view of submitted surveys with edit and delete functionality.
- **CRUD Operations:** Create(this performed from survey-form page), Read, Update, and Delete surveys (are performed from survey-list page where we provided features for update and delete the survey from there) in a structured way.

Frontend:

Home:

home.ts:



home.component.ts U × | survey-list.component.html U | survey-list.component.css U

survey-app > src > app > home > home.component.ts > ...

```typescript
// Meghana Adithi Bongu - G01446886
// Tarun Naga Sai Chadaram - G01445928
// Bhavishya kollipara - G01455074
// Deepika Hemant Tendulkar - G01451861


import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';
import { RouterModule } from '@angular/router'; // For routerLink

@Component({
  selector: 'app-home',
  templateUrl: './home.component.html',
  styleUrls: ['./home.component.css'],
  standalone: true,
  imports: [CommonModule, RouterModule], // Add necessary imports
})
export class HomeComponent {}

```

The HomeComponent is a standalone Angular component defined with the selector app-home, and it includes template and style URLs for rendering. It imports CommonModule for common directives and RouterModule for routing capabilities.

Survey-form.html: This Angular form is a comprehensive survey form for students, featuring fields for personal details, survey date, preferences, and additional comments, all bound to the `survey` model using `ngModel`. It includes various input types like text, date, checkbox, radio, and select, and is designed to handle form submission and resetting via the `onSubmit` and `resetForm` methods.

Survey-form.css: The .survey-form-container class styles the survey form with a light gold background, green border, and subtle shadow to create an inviting, professional appearance. Button styles are defined for primary, secondary, success, and info buttons with color transitions to enhance interactivity and visual appeal when hovered.

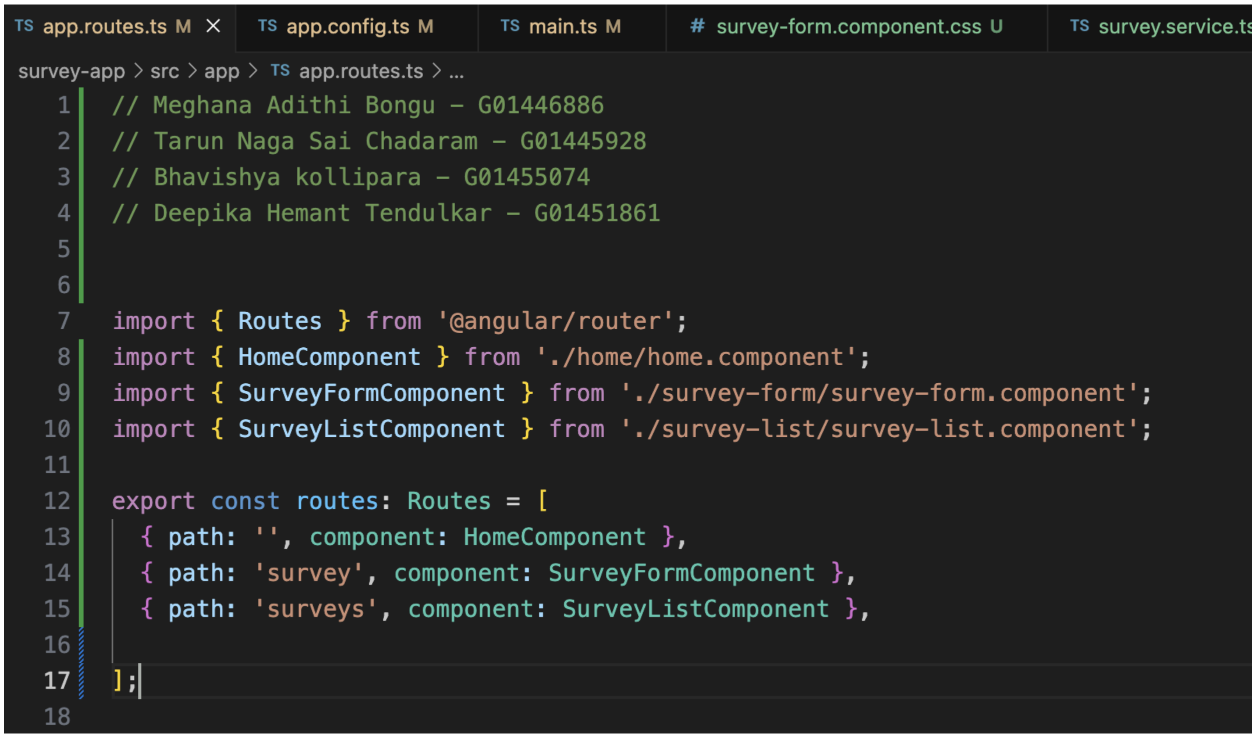
Survey-form.ts: The SurveyFormComponent manages the survey form, handling user input, validation, and submission via the SurveyService. It includes functionality to toggle the selected "liked most" options, validate required fields, and reset the form after submission, with appropriate alerts for success or failure.

Survey-list.html: The SurveyListComponent displays a list of surveys with options to view, edit, or delete each survey. Users can toggle between view and edit modes, make changes, and save or cancel the edits, along with navigating to fill out a new survey or return to the homepage.

Survey-list.css: The provided CSS styles enhance the appearance of a survey card, including its border, background color, and hover effects, making it visually appealing with smooth transitions. It also defines custom styles for various button types (primary, danger, success, secondary) with specific colors and hover effects to improve user interaction.submission, resetting to the home page and survey list using routerLink.

Survey-list.ts: The SurveyListComponent manages the display and editing of survey data, allowing users to view, edit, and delete surveys through interaction with the SurveyService. It fetches survey data on initialization, handles survey editing, and updates or deletes surveys via HTTP requests, with appropriate success and error handling.

app-routes.ts:

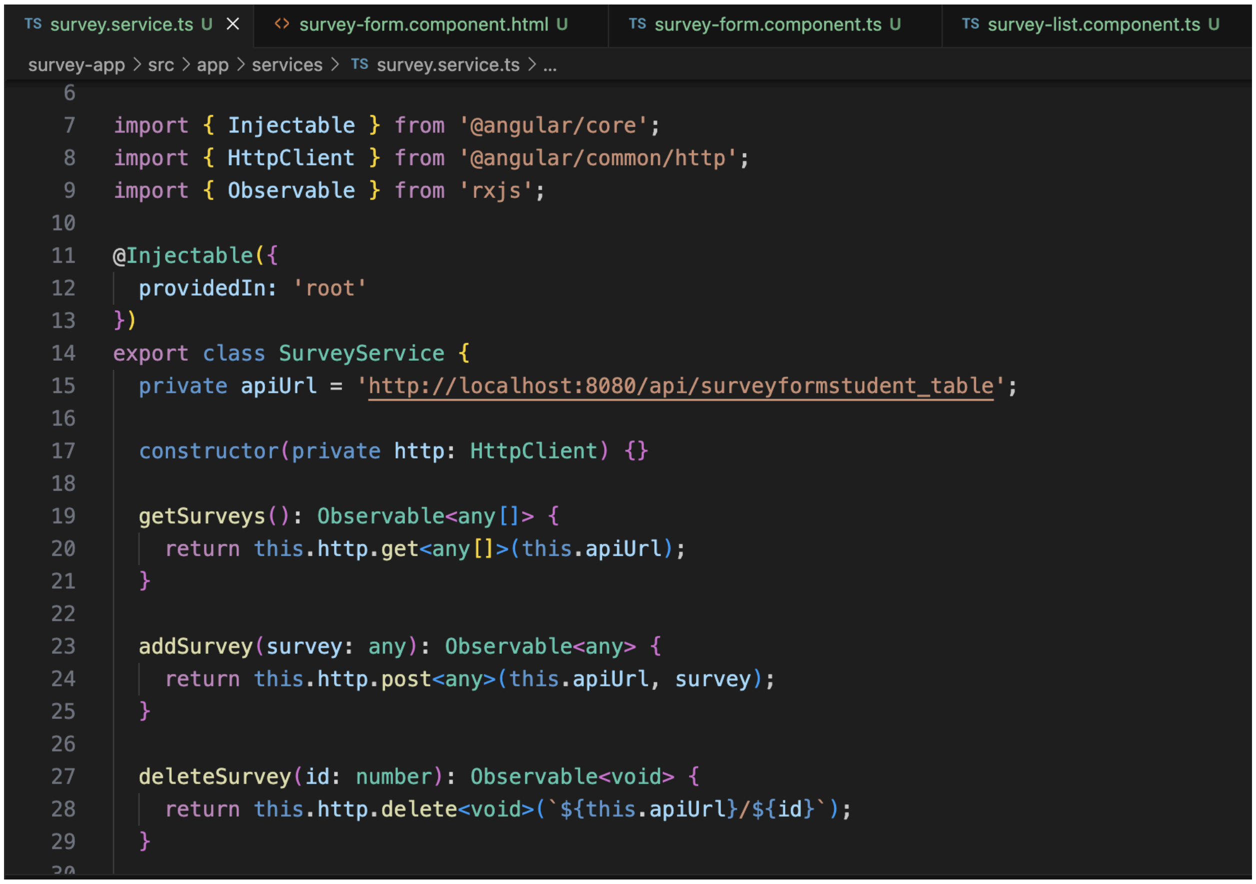


TS app.routes.ts M
TS app.config.ts M
TS main.ts M
# survey-form.component.css U
TS survey.service.ts
survey-app > src > app > TS app.routes.ts > ...

```
// Meghana Adithi Bongu - G01446886
// Tarun Naga Sai Chadaram - G01445928
// Bhavishya kollipara - G01455074
// Deepika Hemant Tendulkar - G01451861


import { Routes } from '@angular/router';
import { HomeComponent } from './home/home.component';
import { SurveyFormComponent } from './survey-form/survey-form.component';
import { SurveyListComponent } from './survey-list/survey-list.component';

export const routes: Routes = [
  { path: '', component: HomeComponent },
  { path: 'survey', component: SurveyFormComponent },
  { path: 'surveys', component: SurveyListComponent },

];

```

The app-routes.ts file defines the routing configuration for the Angular application, mapping specific paths to their respective components. It includes routes for the home page, survey form page, and survey list page, enabling navigation between these views.

Survey.service files:

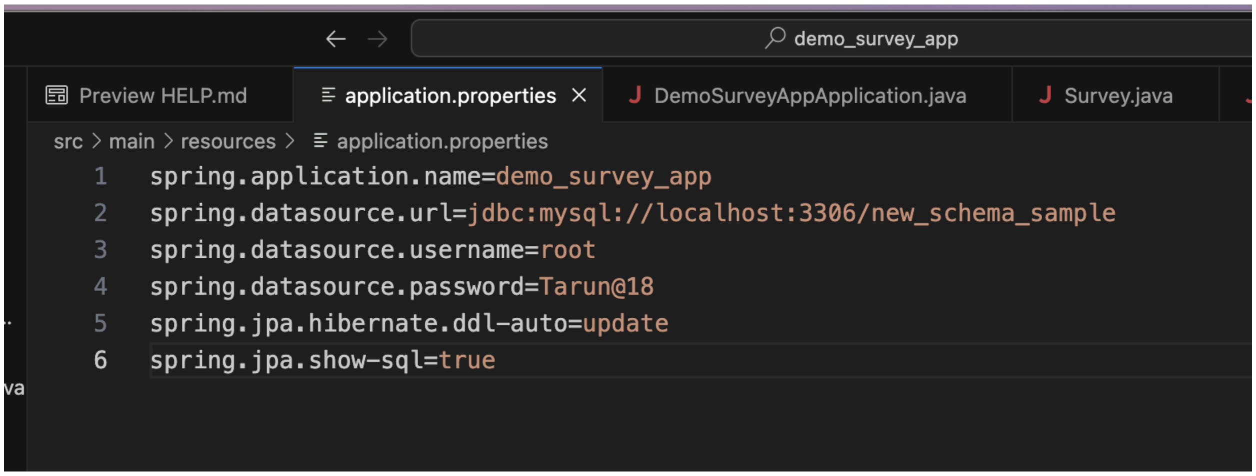

TS survey.service.ts U
survey-form.component.html U
TS survey-form.component.ts U
TS survey-list.component.ts U
survey-app > src > app > services > TS survey.service.ts > ...

```typescript

import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root'
})
export class SurveyService {
  private apiUrl = 'http://localhost:8080/api/surveyformstudent_table';

  constructor(private http: HttpClient) {}

  getSurveys(): Observable<any[]> {
    return this.http.get<any[]>(this.apiUrl);
  }

  addSurvey(survey: any): Observable<any> {
    return this.http.post<any>(this.apiUrl, survey);
  }

  deleteSurvey(id: number): Observable<void> {
    return this.http.delete<void>(`${this.apiUrl}/${id}`);
  }
```

The SurveyService is an Angular service that provides methods to interact with the survey API. It includes functionality to fetch surveys (getSurveys), add a new survey (addSurvey), delete a survey by its ID (deleteSurvey), and update an existing survey (updateSurvey).

Backend:

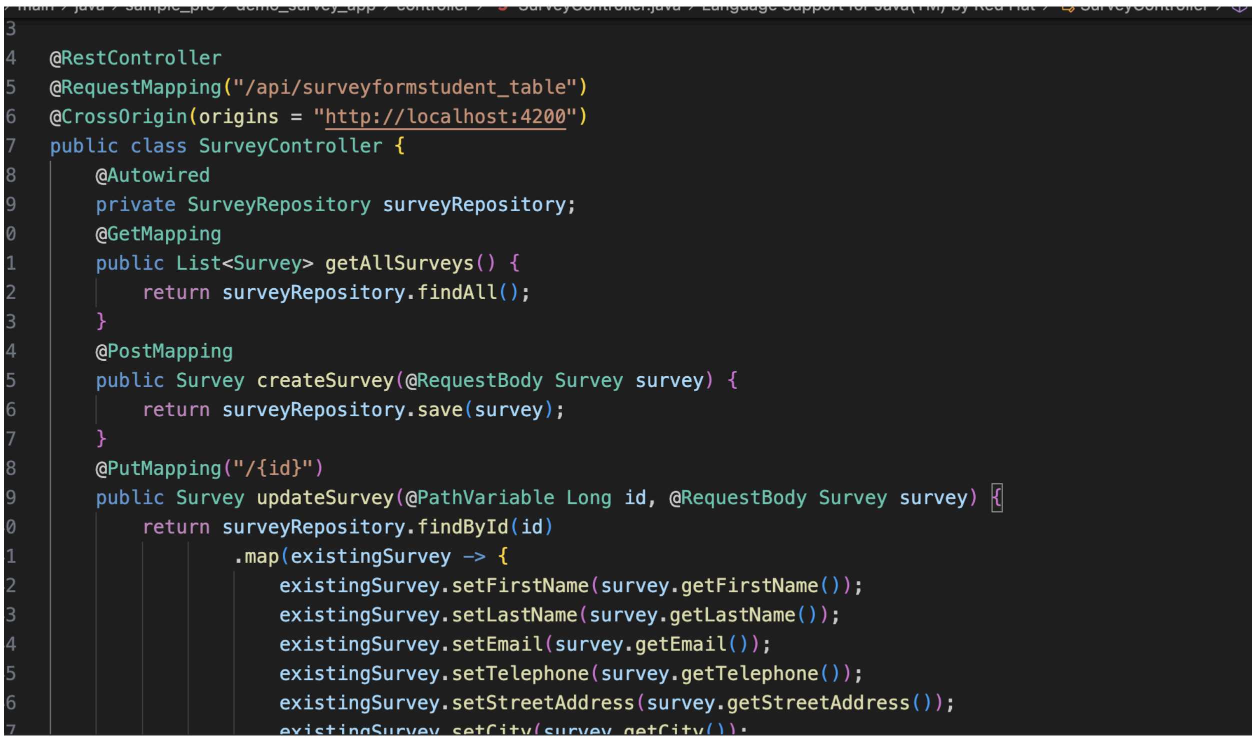
Application.properties:


demo_survey_app
Preview HELP.md
application.properties
DemoSurveyAppApplication.java
Survey.java
src > main > resources > application.properties

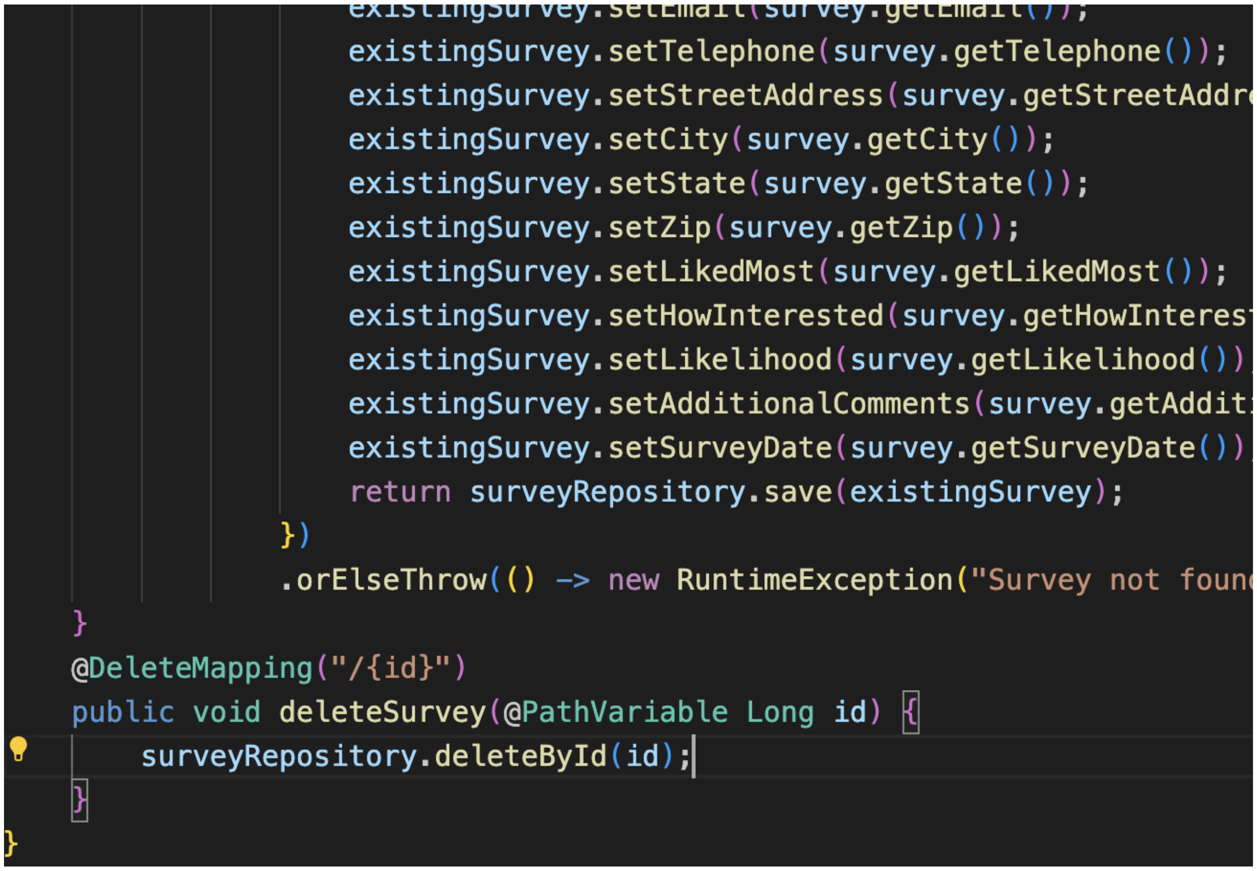
```
spring.application.name=demo_survey_app
spring.datasource.url=jdbc:mysql://localhost:3306/new_schema_sample
spring.datasource.username=root
spring.datasource.password=Tarun@18
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

The application.properties file configures database connection details and enables automatic schema updates with Hibernate, while also logging SQL queries.

SurveyController.java:



```java
@RestController
@RequestMapping("/api/surveyformstudent_table")
@CrossOrigin(origins = "http://localhost:4200")
public class SurveyController {
    @Autowired
    private SurveyRepository surveyRepository;
    @GetMapping
    public List<Survey> getAllSurveys() {
        return surveyRepository.findAll();
    }
    @PostMapping
    public Survey createSurvey(@RequestBody Survey survey) {
        return surveyRepository.save(survey);
    }
    @PutMapping("/{id}")
    public Survey updateSurvey(@PathVariable Long id, @RequestBody Survey survey) {
        return surveyRepository.findById(id)
                .map(existingSurvey -> {
                    existingSurvey.setFirstName(survey.getFirstName());
                    existingSurvey.setLastName(survey.getLastName());
                    existingSurvey.setEmail(survey.getEmail());
                    existingSurvey.setTelephone(survey.getTelephone());
                    existingSurvey.setStreetAddress(survey.getStreetAddress());
                    existingSurvey.setCity(survey.getCity());
```

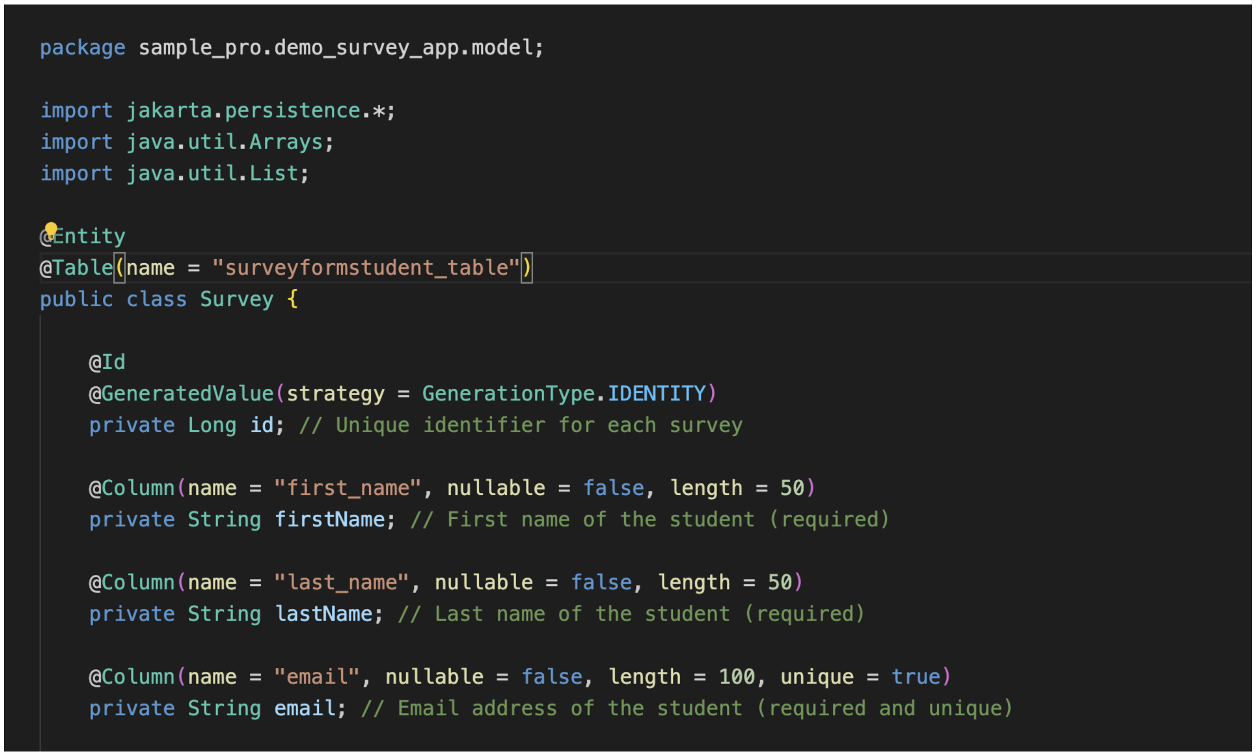


```
                existingSurvey.setEmail(survey.getEmail());
                existingSurvey.setTelephone(survey.getTelephone());
                existingSurvey.setStreetAddress(survey.getStreetAddr
                existingSurvey.setCity(survey.getCity());
                existingSurvey.setState(survey.getState());
                existingSurvey.setZip(survey.getZip());
                existingSurvey.setLikedMost(survey.getLikedMost());
                existingSurvey.setHowInterested(survey.getHowIntere
                existingSurvey.setLikelihood(survey.getLikelihood())
                existingSurvey.setAdditionalComments(survey.getAddit
                existingSurvey.setSurveyDate(survey.getSurveyDate())
                return surveyRepository.save(existingSurvey);
            })
            .orElseThrow(() -> new RuntimeException("Survey not foun
    }
    @DeleteMapping("/{id}")
    public void deleteSurvey(@PathVariable Long id) {
        surveyRepository.deleteById(id);
    }
}
```

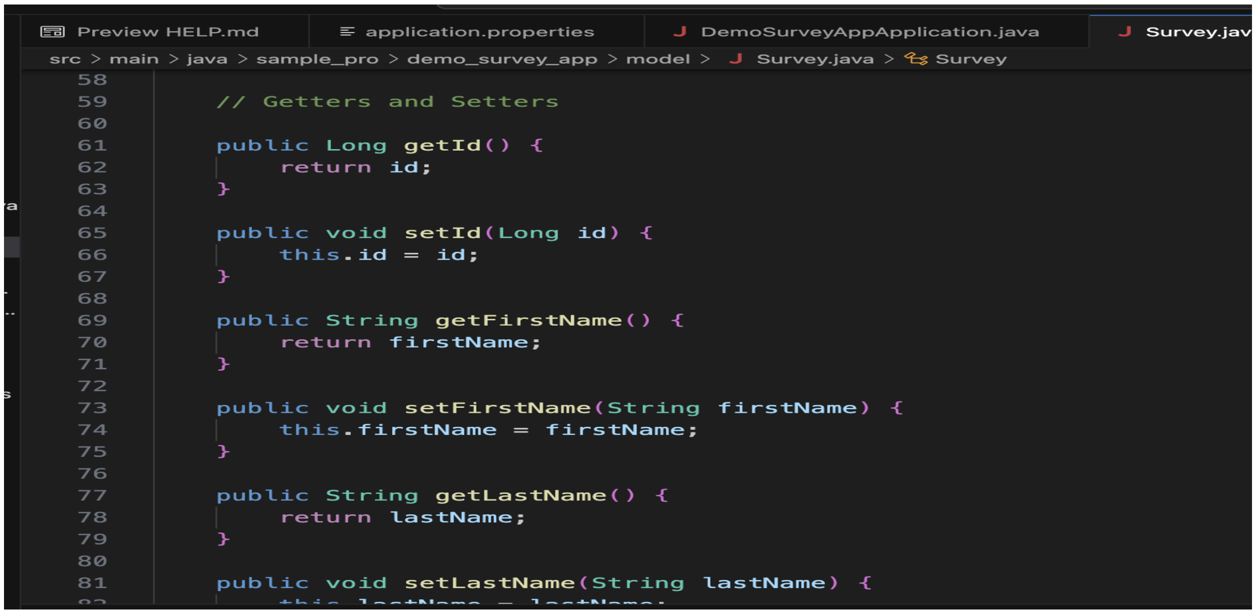
The SurveyController is a Spring Boot REST API that provides CRUD operations for the Survey entity. It supports fetching, creating, updating, and deleting surveys with cross-origin requests allowed from http://localhost:4200.

Model:

Survey.java:



```java
package sample_pro.demo_survey_app.model;

import jakarta.persistence.*;
import java.util.Arrays;
import java.util.List;

@Entity
@Table(name = "surveyformstudent_table")
public class Survey {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id; // Unique identifier for each survey

    @Column(name = "first_name", nullable = false, length = 50)
    private String firstName; // First name of the student (required)

    @Column(name = "last_name", nullable = false, length = 50)
    private String lastName; // Last name of the student (required)

    @Column(name = "email", nullable = false, length = 100, unique = true)
    private String email; // Email address of the student (required and unique)
```

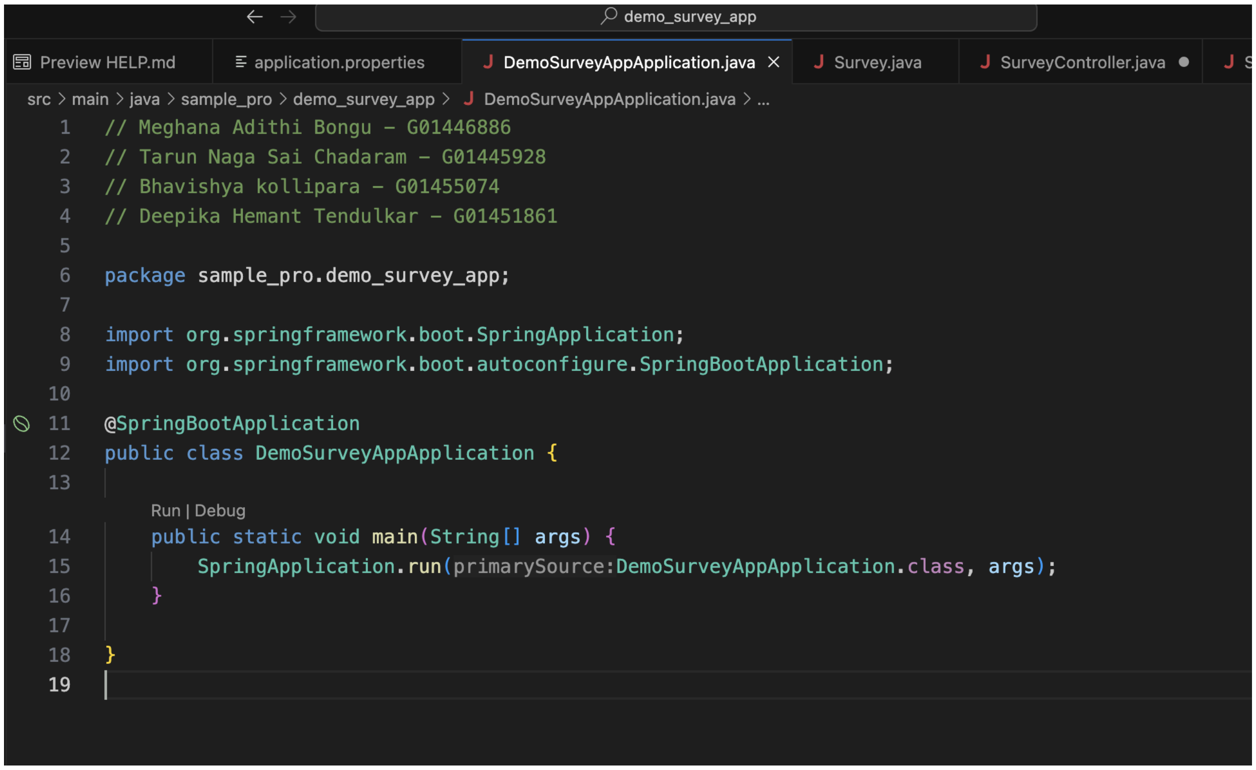

Preview HELP.md
application.properties
DemoSurveyAppApplication.java
Survey.jav
src > main > java > sample_pro > demo_survey_app > model > Survey.java > Survey

```java

    // Getters and Setters

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public void setLastName(String lastName) {
        this.lastName = lastName;
```

This Survey class represents a survey record with fields for the student's personal details, survey responses, and survey submission date. It uses JPA annotations to map the class to the surveyformstudent_table in the database and includes custom getter and setter methods for handling lists, such as likedMost, which is stored as a comma-separated string.

Repository:

DemoSurevyAppApplication.java:


demo_survey_app
Preview HELP.md
application.properties
DemoSurveyAppApplication.java
Survey.java
SurveyController.java
src > main > java > sample_pro > demo_survey_app > DemoSurveyAppApplication.java > ...
Run | Debug

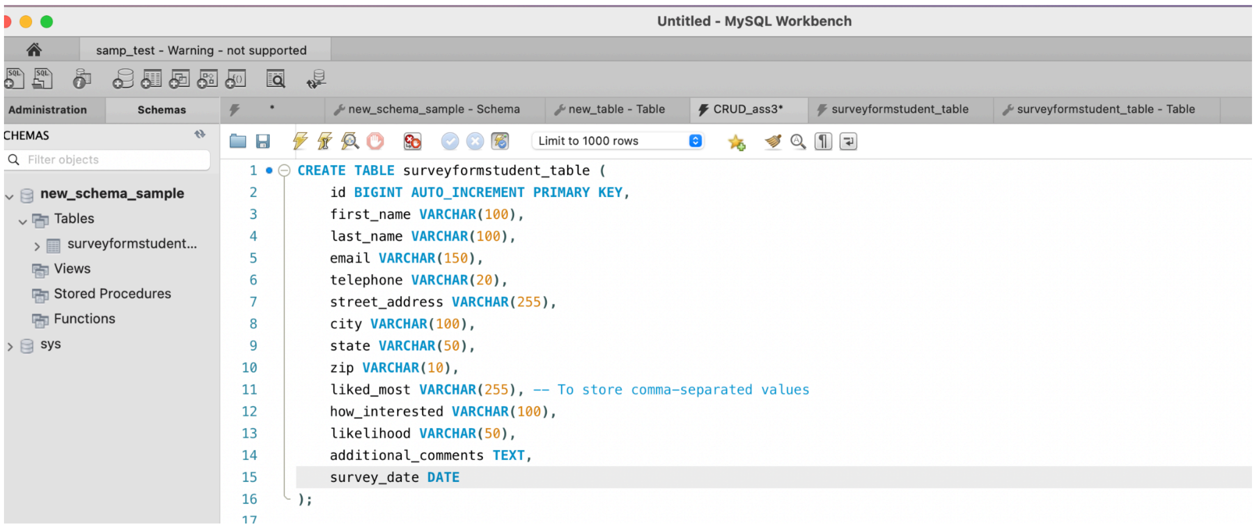
```java
// Meghana Adithi Bongu - G01446886
// Tarun Naga Sai Chadaram - G01445928
// Bhavishya kollipara - G01455074
// Deepika Hemant Tendulkar - G01451861

package sample_pro.demo_survey_app;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoSurveyAppApplication {

    public static void main(String[] args) {
        SpringApplication.run(primarySource:DemoSurveyAppApplication.class, args);
    }

}
```

The DemoSurveyAppApplication class is the entry point of the Spring Boot application, containing the main method that runs the application. It is annotated with @SpringBootApplication, which enables component scanning, auto-configuration, and configuration properties in the Spring context.

Database:

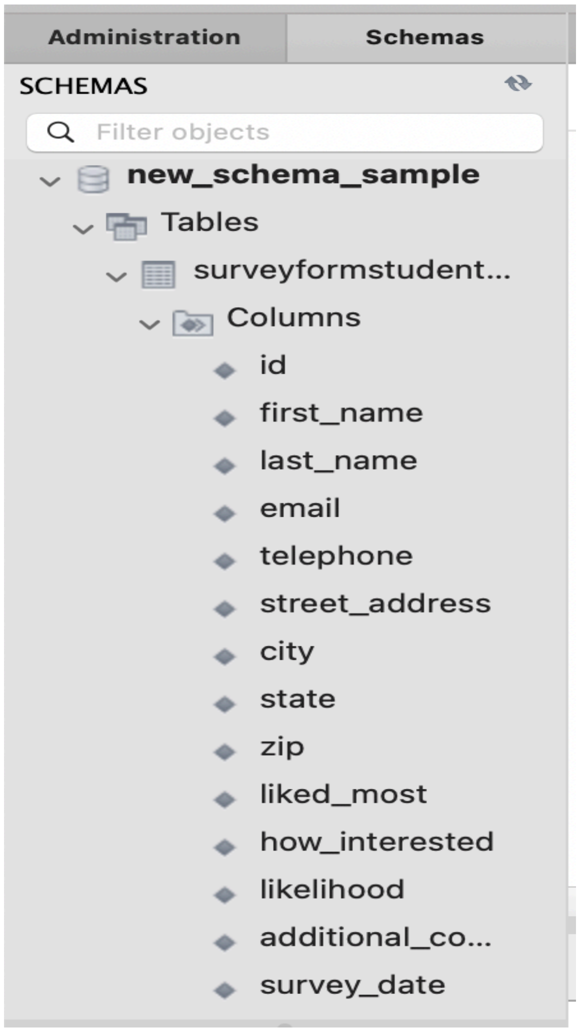
We have carried out the database setup through SQL server and SQL Workbench.

The table creation and schema are as follows:

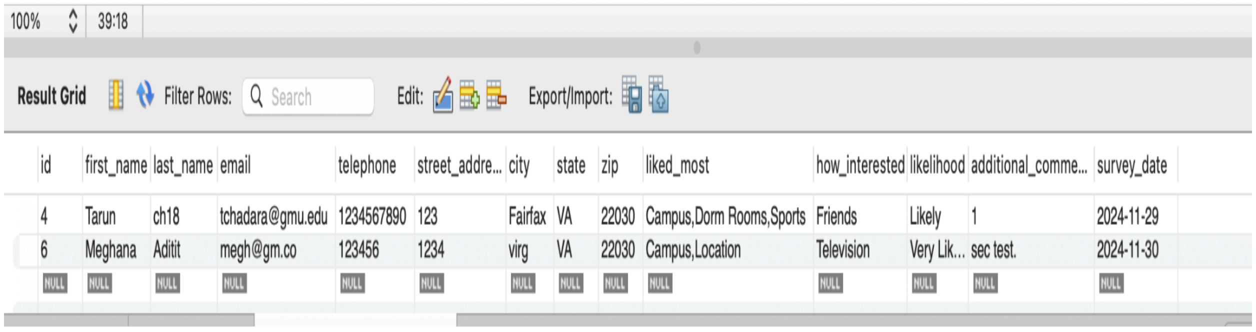


```sql
CREATE TABLE surveyformstudent_table (
    id BIGINT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(100),
    last_name VARCHAR(100),
    email VARCHAR(150),
    telephone VARCHAR(20),
    street_address VARCHAR(255),
    city VARCHAR(100),
    state VARCHAR(50),
    zip VARCHAR(10),
    liked_most VARCHAR(255), -- To store comma-separated values
    how_interested VARCHAR(100),
    likelihood VARCHAR(50),
    additional_comments TEXT,
    survey_date DATE
);
```

These are the columns formed:



The table will be formed as below and the data will be flown in and out from web page to here from this table:



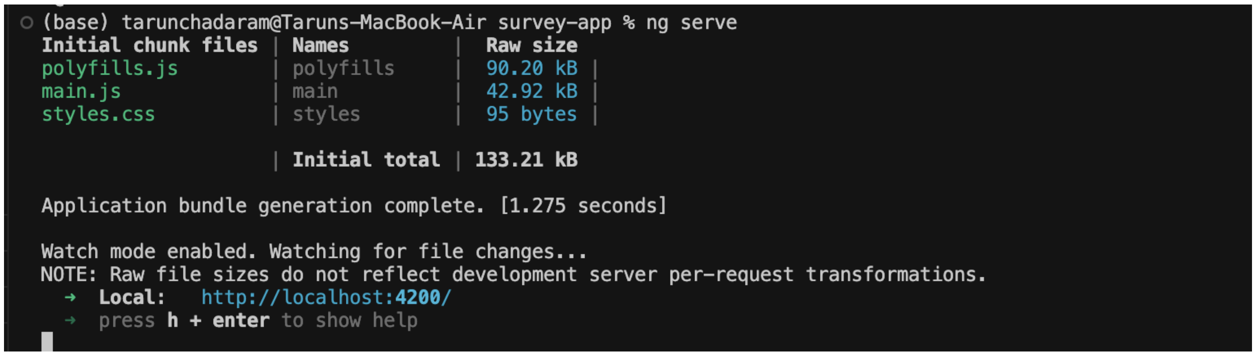
100% 39:18

Result Grid Filter Rows: Search Edit: Export/Import:

	id	first_name	last_name	email	telephone	street_addre...	city	state	zip	liked_most	how_interested	likelihood	additional_comme...	survey_date
	4	Tarun	ch18	tchadara@gmu.edu	1234567890	123	Fairfax	VA	22030	Campus,Dorm Rooms,Sports	Friends	Likely	1	2024-11-29
	6	Meghana	Aditit	megh@gm.co	123456	1234	virg	VA	22030	Campus,Location	Television	Very Lik...	sec test.	2024-11-30
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

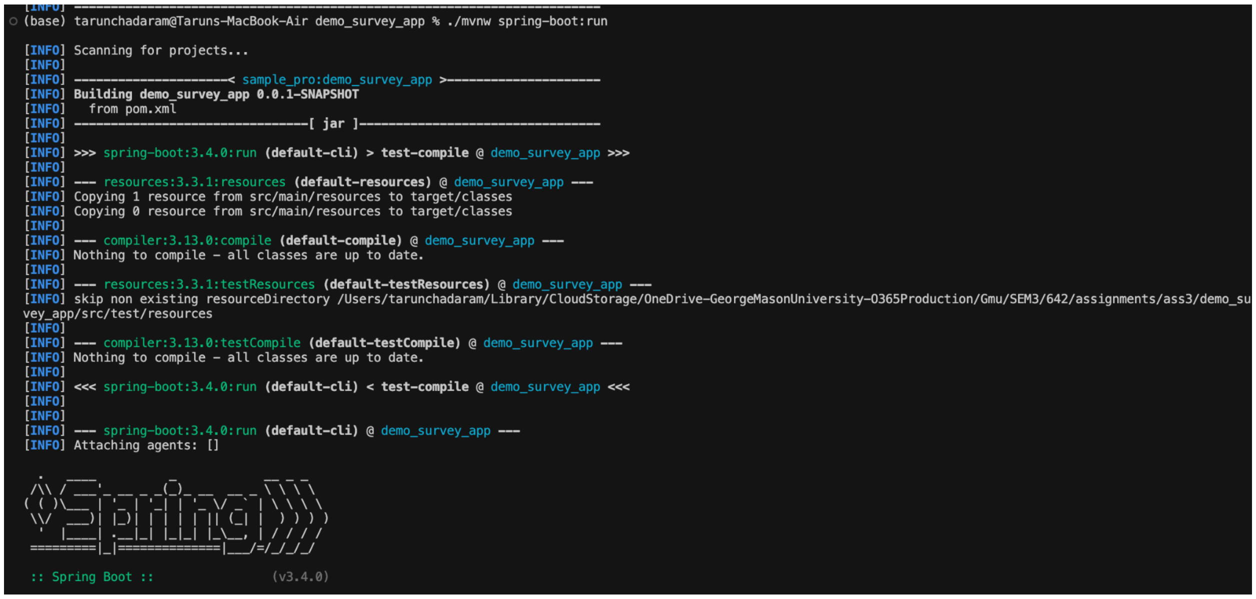
Running and final outputs:

Command:
Running the frontend part (angular part):
ng serve



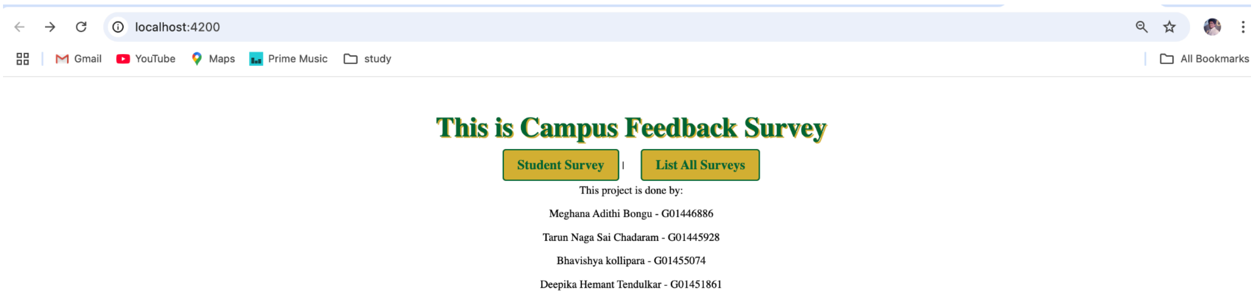
```
○ (base) tarunchadaram@Taruns-MacBook-Air survey-app % ng serve
  Initial chunk files | Names         |  Raw size
  polyfills.js        | polyfills     |  90.20 kB |
  main.js             | main          |  42.92 kB |
  styles.css          | styles        |  95 bytes |

                      | Initial total | 133.21 kB

  Application bundle generation complete. [1.275 seconds]

  Watch mode enabled. Watching for file changes...
  NOTE: Raw file sizes do not reflect development server per-request transformations.
    ➜  Local:   http://localhost:4200/
    ➜  press h + enter to show help
```

Running the Backend part (in spring boot):
./mvnw spring-boot:run



```
[INFO] ------------------------------------------------------------------------
○ (base) tarunchadaram@Taruns-MacBook-Air demo_survey_app % ./mvnw spring-boot:run

[INFO] Scanning for projects...
[INFO]
[INFO] --------------------< sample_pro:demo_survey_app >---------------------
[INFO] Building demo_survey_app 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] >>> spring-boot:3.4.0:run (default-cli) > test-compile @ demo_survey_app >>>
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ demo_survey_app ---
[INFO] Copying 1 resource from src/main/resources to target/classes
[INFO] Copying 0 resource from src/main/resources to target/classes
[INFO]
[INFO] --- compiler:3.13.0:compile (default-compile) @ demo_survey_app ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ demo_survey_app ---
[INFO] skip non existing resourceDirectory /Users/tarunchadaram/Library/CloudStorage/OneDrive-GeorgeMasonUniversity-0365Production/Gmu/SEM3/642/assignments/ass3/demo_su
vey_app/src/test/resources
[INFO]
[INFO] --- compiler:3.13.0:testCompile (default-testCompile) @ demo_survey_app ---
[INFO] Nothing to compile - all classes are up to date.
[INFO]
[INFO] <<< spring-boot:3.4.0:run (default-cli) < test-compile @ demo_survey_app <<<
[INFO]
[INFO]
[INFO] --- spring-boot:3.4.0:run (default-cli) @ demo_survey_app ---
[INFO] Attaching agents: []

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.4.0)
```

Home:



localhost:4200

This is Campus Feedback Survey

Student Survey | List All Surveys

This project is done by:

Meghana Adithi Bongu - G01446886

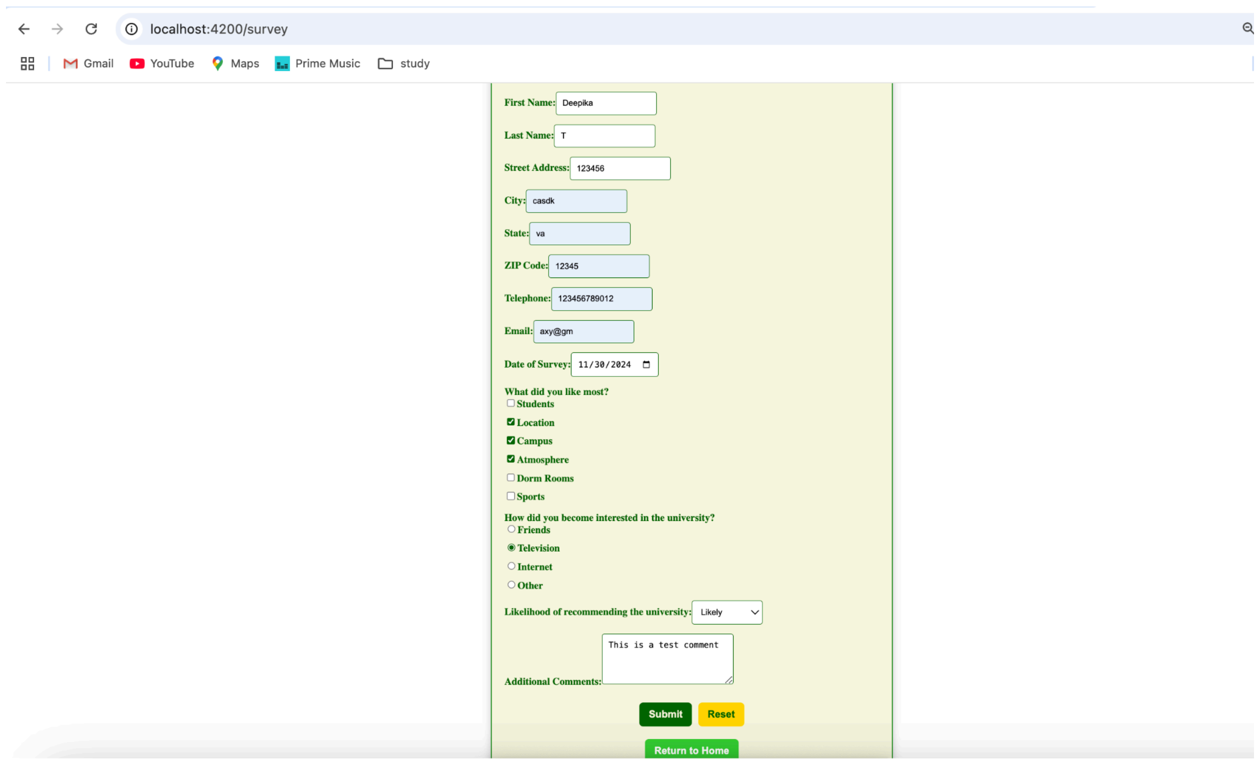
Tarun Naga Sai Chadaram - G01445928

Bhavishya kollipara - G01455074

Deepika Hemant Tendulkar - G01451861

Survey-form:

The final output of the survey form application's in the below snips. With respective fields like first name, last name, street address, city, state, zip, telephone number, e-mail, and date of survey. It has submit, cancel, return to home and list of surveys buttons.



First Name: Deepika

Last Name: T

Street Address: 123456

City: casdk

State: va

ZIP Code: 12345

Telephone: 123456789012

Email: axy@gm

Date of Survey: 11/30/2024

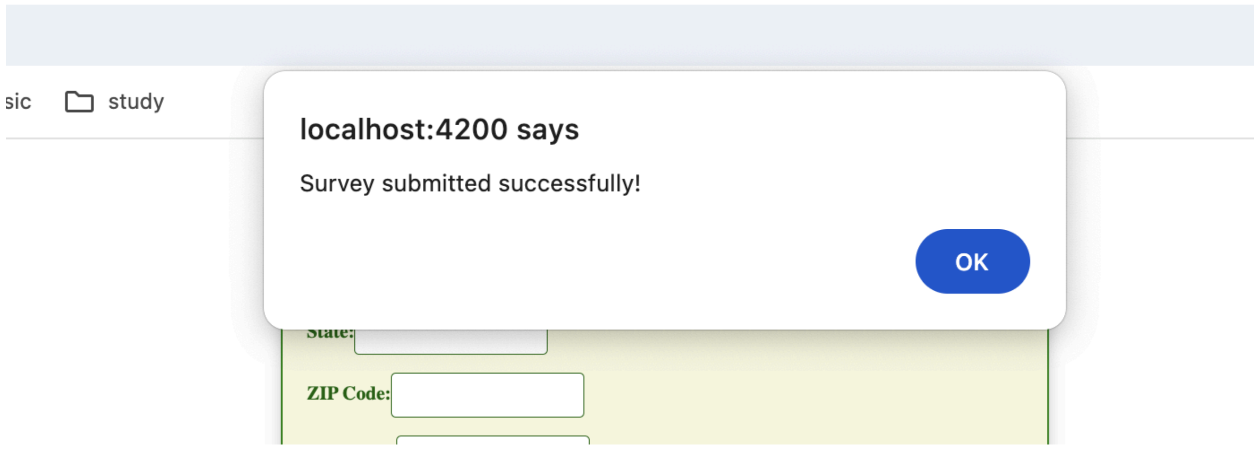
What did you like most?
- [ ] Students
- [x] Location
- [x] Campus
- [x] Atmosphere
- [ ] Dorm Rooms
- [ ] Sports

How did you become interested in the university?
Friends [✓ Television] Internet Other

Likelihood of recommending the university: Likely

Additional Comments: This is a test comment

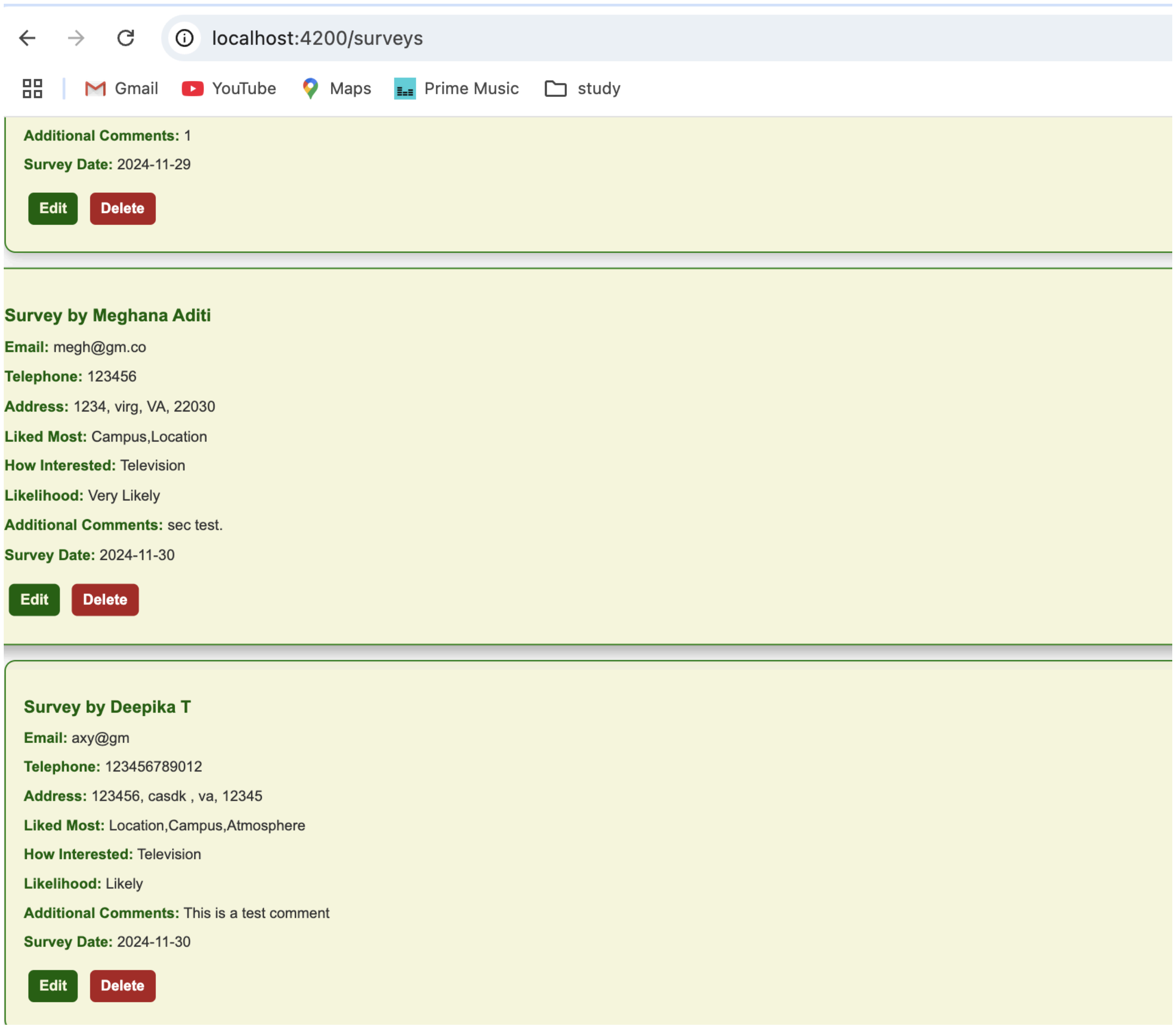
The survey gets submitted once we give the required fields of the form and click submit with this message displayed.



localhost:4200 says

Survey submitted successfully!

Survey-list:
Below are the submitted responses.



Additional Comments: 1

Survey Date: 2024-11-29

Survey by Meghana Aditi

Email: megh@gm.co

Telephone: 123456

Address: 1234, virg, VA, 22030

Liked Most: Campus,Location

How Interested: Television

Likelihood: Very Likely

Additional Comments: sec test.

Survey Date: 2024-11-30

Survey by Deepika T

Email: axy@gm

Telephone: 123456789012

Address: 123456, casdk , va, 12345

Liked Most: Location,Campus,Atmosphere

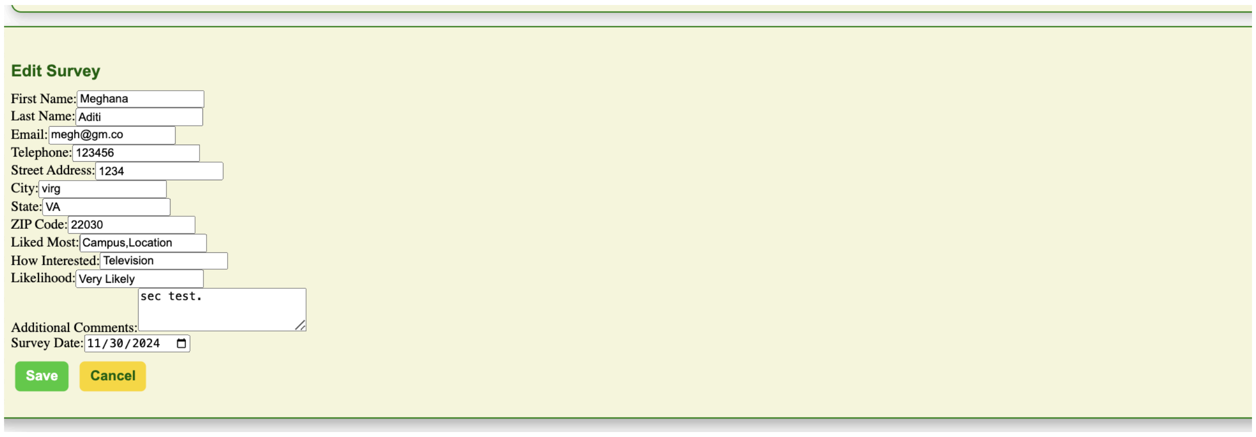
How Interested: Television

Likelihood: Likely

Additional Comments: This is a test comment

Survey Date: 2024-11-30

Editing in survey list:
There are edit and delete options given as well to edit any of the following submitted fields and delete the whole response at once.



Edit Survey

First Name:	Meghana
Last Name:	Aditi
Email:	megh@gm.co
Telephone:	123456
Street Address:	1234
City:	virg
State:	VA
ZIP Code:	22030
Liked Most:	Campus,Location
How Interested:	Television
Likelihood:	Very Likely
Additional Comments:	sec test.
Survey Date:	11/30/2024