

The First Tee Documentation

by

Ediberto Cruz Cruz, Jose Cortez & Morgan Juran
Mentor: Dr. Lara

Senior Capstone

Spring 2017

CALIFORNIA STATE UNIVERSITY MONTEREY BAY



SEASIDE, CALIFORNIA

Project Documentation

[Introduction](#)

[Section 1: User Documentation](#)

[Home](#)

[Reports](#)

[Add Student](#)

[Charts](#)

[Section 2: Django](#)

[Templates](#)

[Views](#)

[Form](#)

[Database Tables](#)

[Section 3: Reports and Charts](#)

[Reports](#)

[Charts](#)

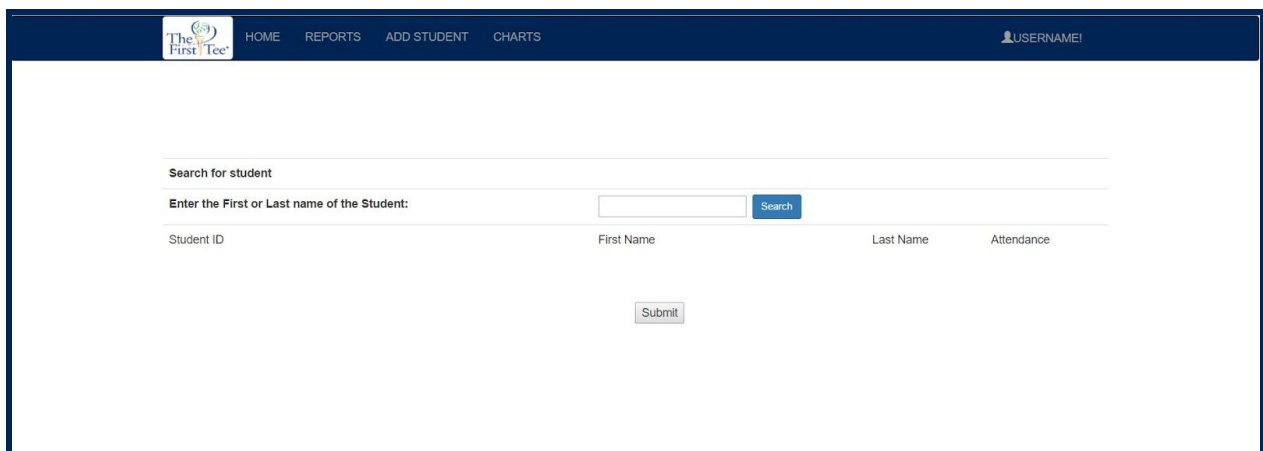
Introduction

First Tee of Monterey County is an after-school program with the goal of giving students a safe place to enhance their learning through tutoring and extracurricular activities. It is part of a larger, national organization, that emphasizes that golf is a “game for everyone,” while teaching students dedication and skills to be used on the course, in the classroom, and in life.

We decided to work on this project after Jose worked at First Tee for his Service Learning. He saw an opportunity to help an organization that reaches thousands of lives, and we all wanted to do something to give back and make our project meaningful to those who take advantage of this program.

Section 1:

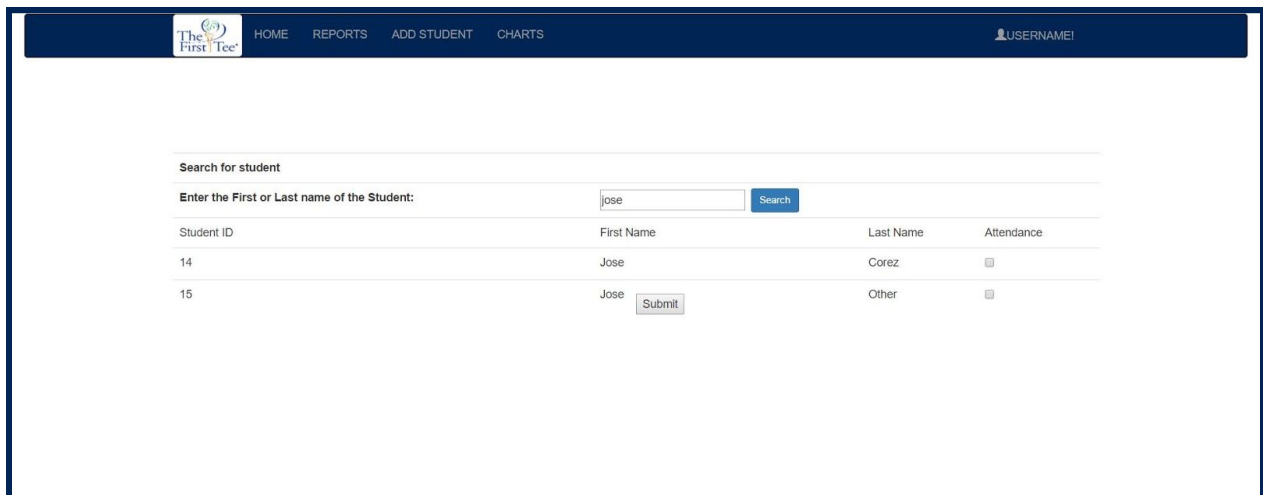
Home



The screenshot shows the home page of a web application for 'The First Tee'. The header is dark blue with the logo on the left and navigation links (HOME, REPORTS, ADD STUDENT, CHARTS) in the center. A user login area on the right shows 'USERNAME!'. The main content area is white and contains a search section with the text 'Search for student' and 'Enter the First or Last name of the Student:'. Below this is a search bar with a 'Search' button. Underneath the search bar is a table with four columns: 'Student ID', 'First Name', 'Last Name', and 'Attendance'. A 'Submit' button is centered below the table.

Student ID	First Name	Last Name	Attendance
------------	------------	-----------	------------

This is the page first visible upon loading the site, also known as the home page. This page also allows users to search for a particular student, or students, as they enter the office. To search for a student, click into the search box, type their name, and click “Search.” This will give a list of students with that name, and will allow attendance to be recorded for that student.




The screenshot shows the home page of 'The First Tee' web application. The header is dark blue with the logo on the left and navigation links (HOME, REPORTS, ADD STUDENT, CHARTS) in the center. A user profile icon and 'USERNAME!' are on the right. The main content area has a search section titled 'Search for student' with a text input field containing 'jose' and a blue 'Search' button. Below this is a table with columns: Student ID, First Name, Last Name, and Attendance. The table contains two rows: one for Student ID 14 with First Name 'Jose' and Last Name 'Corez', and another for Student ID 15 with First Name 'Jose' and Last Name 'Other'. Each row has an attendance checkbox. A 'Submit' button is located below the table.

Student ID	First Name	Last Name	Attendance
14	Jose	Corez	<input type="checkbox"/>
15	Jose	Other	<input type="checkbox"/>


This is what the page looks like with student information populated, and the attendance box to the right of their name. When that box is checked and the “submit” button is clicked, the timestamp is automatically sent to the database, and the next student can be checked in. Though we did not get to the timestamp implementation that foundation for it is there.

Reports

 HOME REPORTS ADD STUDENT CHARTS USERNAME!			
Students and School Reports			Students and School Reports
School Name	Number of Students	Gender	
Alisal High	1	Female	
Alisal High	1	Male	
Alvarez High	1	Male	
CSUMB	2	Female	
CSUMB	3	Male	
North Salinas High	1	Male	
Salinas High	2	Male	
School of Testing	1	Male	
School Name	Number of Students	Ethnicity	
Alisal High	1		
Alisal High	1	Black or African-Ame	
Alvarez High	1		
CSUMB	3	Black or African-Ame	
CSUMB	2	Latino/Hispanic	
North Salinas High	1	Latino/Hispanic	
Salinas High	1		
Salinas High	1	Latino/Hispanic	
School of Testing	1	Hispanic	

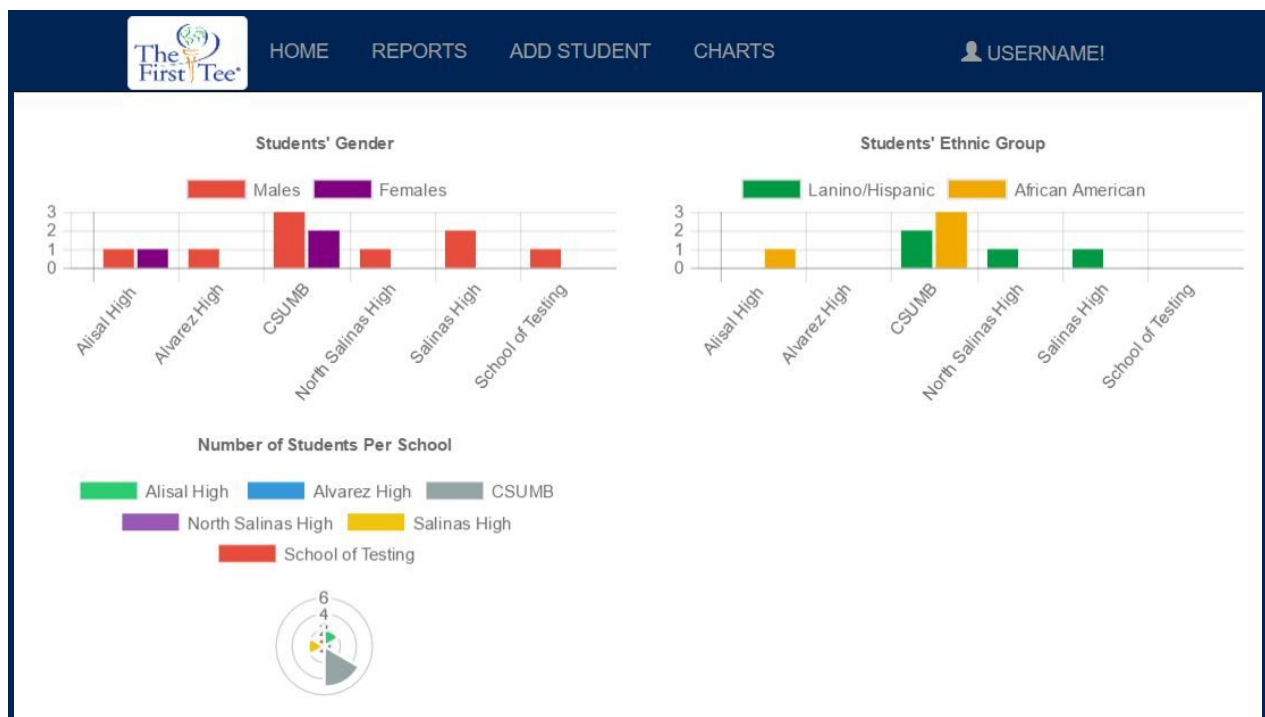
The next tab across the top is “Reports.” This page shows the textual report of demographic breakdowns pertaining to the program. For now, this page shows the breakdown of ethnicity and gender at the participating schools.

Add Student

 HOME REPORTS ADD STUDENT CHARTS USERNAME!	
ID:	<input type="text"/>
First Name:	<input type="text"/>
Last Name:	<input type="text"/>
Phone Number:	<input type="text"/>
Student grade:	<input type="text" value="1"/>
DOB:	<input type="text"/>
Student sex:	<input type="text" value="Male"/>
School:	<input type="text"/>
Note:	<input type="text"/>
Student level:	<input type="text" value="Target"/>
Student ethnicities:	<input type="text" value="Black or African-American"/>
<input type="button" value="Add Student"/>	

One more tab over shows “Add Student.” This is the tab administrators will have access to when they sign in and need to add a new student to the program. The page asks for the student ID, their first and last names, phone number, current grade, date of birth, gender, school, student level, and ethnicity, as well as a place to put notes.

Charts



The final tab option shows the graphical representation of the data, in this case, the demographic breakdown of students. The top left chart shows the number of males and females enrolled from each school, while the one on the right shows the ethnic breakdown of the schools. The final chart shows the percent of participation of each school compared to the others. It is

represented in a Polar Area chart, rather than a pie chart, to highlight the difference in the number of students attending First Tee from the surrounding schools.

Section 2: Django

Templates:

The templates folder is where all the html templates are stored and where any new web pages should be added. Any django code added into any of these templates should be surrounded by {% %}.

Views:

The “views.py” file is where you add or edit any views. This is where you you can add a large portion of your backend code (such as database queries, sorts, algorithms). These views also control what gets sent to the templates and what happens when a form submitted. Queries are performed using built in django functions for example,

`Admin.objects.get(admin_id=2017)` will look through the “Admin” table for the row with the id 2017.

Forms:

The “forms.py” file is where we keep all of our current forms. This is where the classes for every form are created. These forms control what fields appear when that function is selected. For example

```
student_id = forms.CharField(label='ID', max_length=20)
```

creates a textfield with a maximum length of 20 characters. Django has methods that help with manipulating their attributes.

Section 3: Report, Charts and Redirecting

On the menu bar there is a reports and charts tap. This part of the document will be focusing on the content and why the information displayed is necessary. Both pages have the same information but is displayed in different form. Unfortunately, not all the information the client wanted is there, so future developers will have to add this for them.

Redirecting Pages

The team that worked on this project had a difficult time redirecting a page using href. However, after long hours of trial and error and extensive research, the team found the solution.

“Myapp” folder contains the “urls.py” file, where all the urls for the pages are located. The first problem was that the name was not included. The following is an example of how it was implemented without success:

```
url(r'^moreInfo/',views.moreIn),
```

The proper way is the following: `url(r'^moreInfo/',views.moreIn, name='moreInfo')`. The name included has to be the name that was given to the HTML document. In this case, the name of the HTML document is “moreInfo.html”.

When the developer wants to use “href” to redirect a page, the following has to be used in the beginning of the block where the redirection code will be: `{% block nav %}`. Do not forget to close your block with: `{% endblock %}`.

In between the “block nav” and the “endblock,” the developer used an unordered list to display the menu. Each list item contains the following:

```
<li>{% block nav-moreInfo %}<a href="{% url 'moreInfo'
%}">REPORTS</a>{% endblock %}</li>.
```

Reports

The “moreInfo.html” file has the reports about the number of females and male for each school. According to the First Tee, there are about 50 different schools and 6000 students. It would be difficult to display the number of schools on a single page but the decision was made to display only the schools with the most number of students.

The “views.py” uses these queries to get the information about the number of males, females and the number of ethnic groups in each school:

```
def moreIn(request):
    form1 = Students.objects.values('school','gender').annotate(count=Count('school')).order_by('school')
    form2 = Students.objects.values('school','ethnicity').annotate(count=Count('school')).order_by('school')
    return render(request,'moreInfo.html',{'form1':form1,'form2':form2})
```

The class “Student” in the “models.py” has information regarding the “about the student” table. The queries above use the “Student” table to retrieve information about students’ gender and ethnic background.

Charts

In the “charts.html” file there are three different charts that display information about the students. Since it is easier to manipulate the queries, the information from them are stored in arrays. The information from “views.py” is sent to “charts.html” by rendering using Python dictionary. However, an array contain strings then extra information is pass to “charts.html”. To avoid sending unneeded information, the following function needs to be used; `simplejson.dumps()`.

If the array has an integer, that function is not necessary. In order to place an array within the `<script>` tag, a variable needs to be assigned the values of the array. The following is an example of how an array that is passed from “views.py” to “charts.html”:

```
<script>
var school = {{ eschool | safe }};
    more info here...
    data: school
```

more info here...

</script>