UCT Mathematics Competition

A very hastily typed README

January 24, 2014

Contents

1 Competition

So I typed this up in 20 min on the last day of my internship. Apolgies to whoever has to read it. Treating the `uct-maths-competition/uctMaths/` as the base directory, I've based the section names and bolded-heading from an `ls`. The excessive use of `'\verb'` is just me being lazy.

1.1 competition/:

- `admin.py`

  Links the logic from `compadmin.py` to the actions that can be taken in the admin interface. This file pretty much defines how each of those admin entries in the Competition section will look - this includes the filters on the right, the actions that can be run and the columns that are shown to the user.

  All the `___Admin(...)` classes inherit from `ImportExportModelAdmin`, as opposed to the usual Django ModelAdmin. This is to allow the Import/Export buttons on the pages. This import/export functionality also requires a resource_class, which is defined in the `models.py`.

  . note:: The SchoolAdmin class has `__str__` and `__unicode__` methods. Use the latter in code (when representing the School object as a string)

- `compadmin.py` (*"competition admin"*)

  This class contains all of the methods that are called by methods in the `admin.py` file. It's length is much to do with the number of lines of code it takes to get the formatting of Excel files right.

- `compadmin_views.py`

  In order to upload the results (RES) files, a new html view was needed. So its functionality is coded here. Much like `views.py`, this handles rendering of the Django template (usually the errors that occur during import)/the POST data that is returned on "Submit."

- `confirmation.py`

  This was actually the first thing I coded. It just generates a text-based email and sends it to the user's email address. There is an option to

cc the competition administrator (which happens when a new entry is submitted or an entry is ammended).

- `forms.py`

  These forms are usually sent to the html rendering methods. They're just the fields that exist in each of the entities. There are a number of widgets that you can tie in here... We didn't really change this file much. Fields that describe the database columns are actually contained in the `models.py` file.

- `__init__.py`

  This just tells `python` that this folder is an entity. It's empty... but important.

- `tests.py`

  We didn't even touch this file.

- `models.py`

  Each of the entities is defined here. SchoolStudents, Venues, etc... The database entries are also defined here -

  eg. `firstname=models.CharField(max_length=255L, db_column='First_name')` defining the firstname of a SchoolStudent object.

- `urls.py`

  When a request for a page comes through, Django uses the regexes in here to find the page. This is also where html urls are bound to methods in `views.py`.

- `views.py`

  Contains the underlying logic for all of the teachers' interface. Includes checking for school association (and redirecting as needed), rendering of html responses, generating forms and saving posted data to the databases. In `urls.py`, these methods are bound to html files and are called when the particular url is called.

  If you're new to Django - this is where it all happens. And my understanding how methods, webpages and redirecting works came from understanding how this file is structured. (ie. how `POST` data works).

1.2 competition/interface:

In the second week we decided to make sure that the teachers' interface was as simple as possible. So most of these html files are actually not even used. The ones that are (and that are interesting).

The html file names also tell you which method in `competition/views.py` they are bound to... but you can always look at `competition/urls.py` if I'm wrong.

- `newstudents.html`

  This is the main entry form. The related JavaScript file in `competition/static/tables.js` is used for entry validation and has control of the Submit button at the bottom. This is the form that is autopopulated if any information has already been entered by the user... which makes the Django template code look quite nasty. (It's not!)

  When the user is redirected to this page (after having selected a school, or clicking the 'Edit entry' button), the form is populated server-side when the template is being filled (with an html `value="..."` tag.)

  When the user has made his/her entry, the form is POST'd back to server when the Submit button is pressed. **The server then deletes all entry information related to that school**... before re-populating the database with the data in the new form.

  This is why validating the form before any data is POST'd back to server is important. When a valid Submit happens, the user and competition admin is snt

- `entry_review.html`

  This actually uses very simmilar logic to that in `newstudents.html` except that no

- `profile.html`

- `school_select.html`

- `submitted.html`

The following two html templates are used in report generation - where the confirmation and school report pdf docuents are set.

- `school_report.html`
- `printer_entry.html`

The logic for `school_report.html` is in `competition/compadmin.py` while the logic for `printer_entry.html` is in `competition/views.py`

1.3 competition/interface/admin:

- `upload_results.html`

  This is where the **uctMaths** app finds the html template for requesting the .RES file from the admin interface. The logic for this is defined `competition/compadmin_views.py`, called from an action defined in `competition/admin.py`