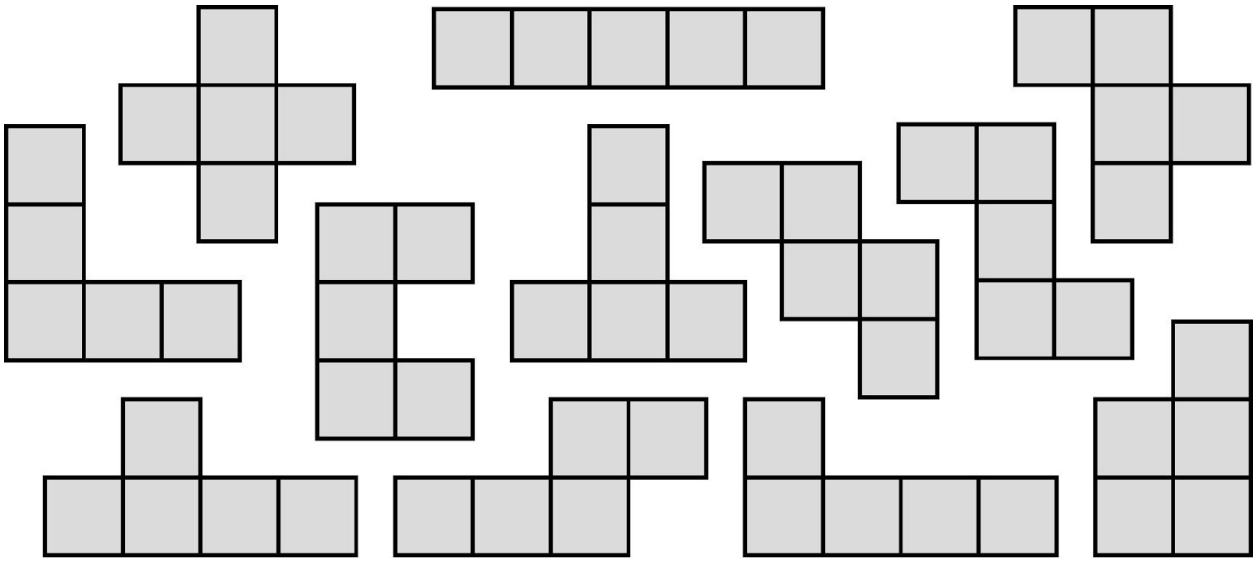
Polyforms

Flash (RIP): Max Zlotskiy, Michael Cheng, Andrew Wong, Gordon Lei
Period 9 Software Development Final Project Design Document



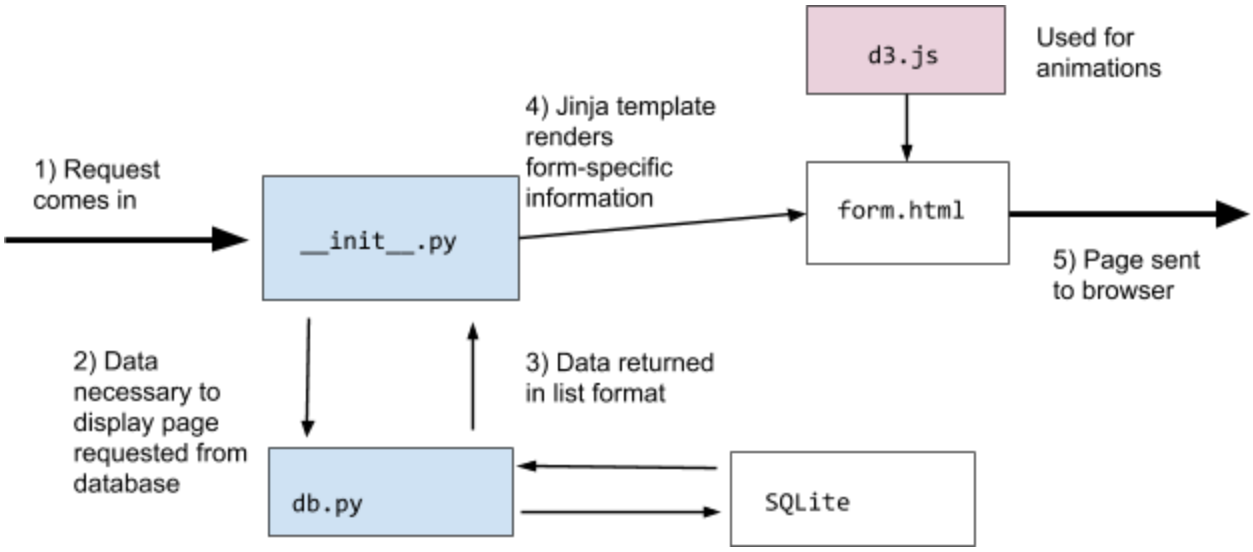
Abstract

Our project is a form creator website, which allows you to create forms(surveys) and gather responses. These responses will be stored in a database. You can then view the responses in a spreadsheet or create charts with them. Our goal, ultimately, is to make a lightweight survey service and integrate really cool data visualization capabilities.

Features

- Registered users can make their own forms/surveys.
- Forms can include different types of questions
    - Radio buttons
    - Text responses
    - Number-only
    - Select all that apply
- the form's creator can share the link to their form to the public. Respondents don't need an account to respond to the form (this can be changed, see below)
- forms that want only one response per person will require respondents to login first
- The form's creator can view everyone's responses in a table or a saveable csv
- A form's creator can limit number of responses

Component Map



Component Description

Front End

- **stylesheet.css** will spruce up the look of the default form input elements
- **graphs.js** will utilize d3 to create graphs based on data found in form results
- **d3.js** will be used to create dynamic charts and graphs
- **index.html** landing page. Link to login. Search bar used to navigate public form results.
- **base.html** a base template to be extended. Contains a menu bar for logged in users
- **form.html** will display a series of questions with input boxes
- **spreadsheet.html** displays form results in a table
- **list.html** used to display forms in a filesystem layout

Routes

Key: underlined routes require login

Route	Description
/	Home page. Has a toolbar with login button. Has options to redirect you to making a form, viewing a form's result, etc. This could also redirect you to a random form to fill out.
/login	Login form.
/my/forms	You can access this page from the toolbar. Has a list of all your creations. Will redirect you to view responses to a form if click on one.

/my/settings	Logged in users can change account settings here. Accessible by toolbar.
/form/respond?id=	The url people go to to fill out a response to the form
/form/edit?id=	This is the editor page where a user makes a new form.
/form/view?id=	Allows the form's creator to see the responses in a table. Graphs made using this data are also displayed on this page.
/ajax	Route that handles the searching through databases for AJAX calls and other calculations. AJAX calls will be used when the creator of the from wants to change how their data results look like (ex. Changing from a bar graph to a pie chart) to avoid having to refresh the page.

- Toolbar that logged in users have at the top of all pages
    - access a list of the forms you created (/my/forms)
    - create a new form (/form/edit)
    - change account settings (/my/settings)
    - logout

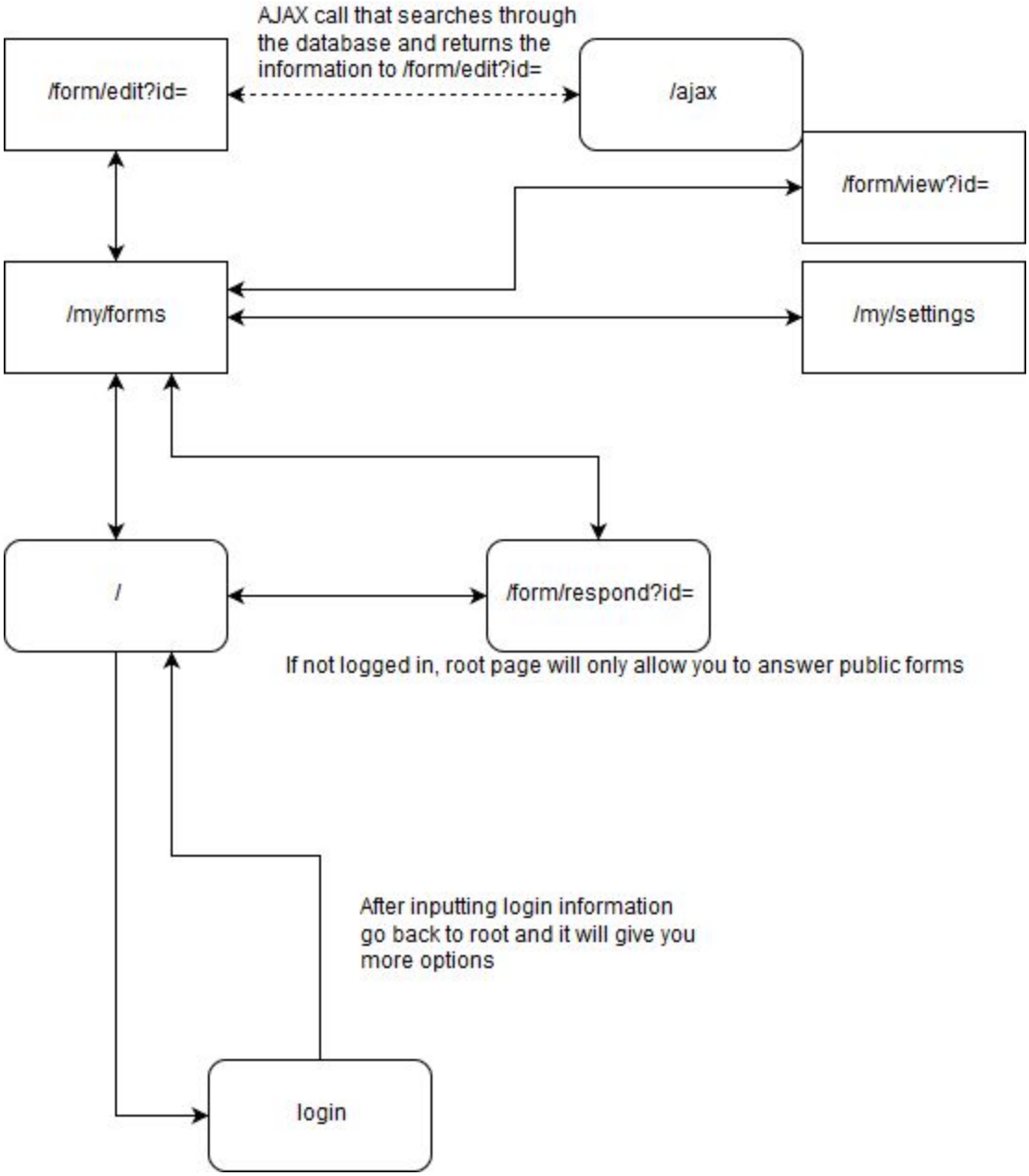
Python Files

- **__init__.py** contains request handling logic used by Flask
- **db.py** for handling saving new forms and graphs into the database; can also pull information from the database and convert it into useful formats (lists, json)
    - `new_form(owner_id, title, list_of_questions)` stores form
    - `get_form(form_id)` returns a list of questions
    - `add_response(form_id, list_of_answers)` stores answers
    - `get_responses(form_id)` returns 2d list of answers

Database (see schema)

- **Accounts**: stores user registration information
- **Forms:** stores metadata about every form that has been created
- **Questions:** stores data about questions (innertext, expected response type, order)
- **Options**: stores choices that will be displayed under radio-button-type questions
- **Responses**: stores all the responses to forms
- **Styles**: stores aesthetic changes made to a spreadsheet
- **Charts** keeps track of which data range a graph uses

Site Map


AJAX call that searches through the database and returns the information to /form/edit?id=
/form/edit?id=
/ajax
/form/view?id=
/my/forms
/my/settings
/
/form/respond?id=
If not logged in, root page will only allow you to answer public forms
After inputting login information go back to root and it will give you more options
login

Database Schema

Here is a summary of our database organization: graphs are attached to responses, which are attached to questions, which are attached to forms, which are attached to the user who created them.

Each database table is represented by a physical table below. The first row has the column names, the second row has the data types, and the last row has example values.

forms Table: stores references to all the forms created as well as who created the forms. The "login_required" option will hold either 0 or 1 to represent True or False. if it is 1, then the person has to login into their account to answer the Form; else they can answer the Form without logging in. The "public_results" option will follow the same "0 or 1 system" where if it equals 0 then only the creator of the form can view the data graphs. If it equals 1 then anyone can view the data graphs. The theme is the color of the html elements.

form_id	**title**	**owner_id**	**login_required**	**public_results**	**theme**
INTEGER PRIMARY KEY	TEXT	INTEGER	INTEGER	INTEGER	TEXT
1	"Fav fruit"	1	0	1	"purple"

responses Table: stores responses to questions. Each record is one answer to one question on one form. When a user submits their responses to a form, their answer to each question is stored here. For a response, the answer to each question shares the same response_id. In the example below, the 25th response to form#1 (which has 2 questions) is shown.

form_id	**question_id**	**user_id**	**response_id**	**response**	**timestamp**
INTEGER	INTEGER	INTEGER	INTEGER	BLOB	TEXT
1	1	1	25	"Not a pineapple"	2008-12-14 03:00
1	2	1	25	"Bob"	2008-12-14 03:00

questions Table: stores all the questions and what kind of question they are. Although all forms share this table, you know which form each question belongs to based on its form id. The order of appearance of the questions is based on question_id (unique for each form, not for the whole table). The type column determines what kind of response this question expects. "Short" is a one-line text response, "long" is a textarea, "number" is a numerical response, and "choice" offers a set of responses (like radio buttons).

question_id	**question**	**type**	**form_id**	**min**	**max**
INTEGER	TEXT	TEXT	INTEGER	INTEGER	INTEGER
The order of appearance of this question on the form	"...?"	"short", "long", "number", "choice"	the id of the form this question belongs to	how many options must be selected, characters typed, or the least value for number questions	how many options can be selected, characters typed, or the top value for number questions

Options Table: stores all the options for "choose your answer" questions (ex. Questions with <select> or radio buttons)

form_id	**question_id**	**option_index**	**text_user_sees**	**value**
INTEGER	INTEGER	INTEGER	TEXT	TEXT
12	1	0	"I like apples"	"apple"

Accounts table: stores the registration information for each user

user_id	**username**	**password**
INTEGER	TEXT	TEXT
1	"pineapple"	"s231dsa321sda321321"

styles table: stores aesthetic settings for the spreadsheet. While every cell has default properties (width, white background), the user can make some changes to them, and those changes are stored here. In the example below, row 1 has a height of 40px.

form_id	row	column	property	value
INTEGER	INTEGER	INTEGER	TEXT	TEXT
1	1	-1	height	40px

Tasks

Task Description	Who's Responsible
project manager	Max
storing form-related things in database	Gordon
database for accounts and charts	Andrew
css styling and layout consistency	Michael
rendering templates(flask) and writing html	Max
flask processing forms	Andrew
creating charts	Michael
javascript for ajax	Gordon