System Design Document

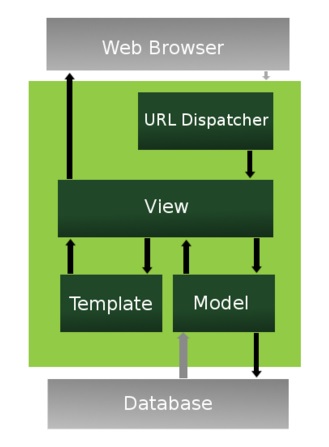
Project: IT6036 Web Security Application Project

By: PAZ

# Architecture

## Web Framework

*Provide an explanation for the django framework. You may reuse the image below, or replace by another one of your choice.*



Framework: A Django framework allows for the quick distribution of said web application as a result of this it allows multiple features and apps joined with Django to be used for the application.

Database: We’ll being using SQlite3 which is bundled with Django which allows for quick installation and able to utilize the tables which within the application.

Template: We’ll being following the DRY methodology (Don’t Repeat Yourself) this allows for multiple views to be created without writing the same code again and again as a result it’s more efficient.

View: Function and class based views used to read html files

URL Dispatcher: We’ll be using the url’s with no framework limitations since Python has a module known as URLconf (URL configuration), as a result this is uses mapping between the URL path to the views so then you’ll be able to see the page.

Web Browser: This displays the web content to the user at the end so they’re able to see what it looks like one put together and functioning.

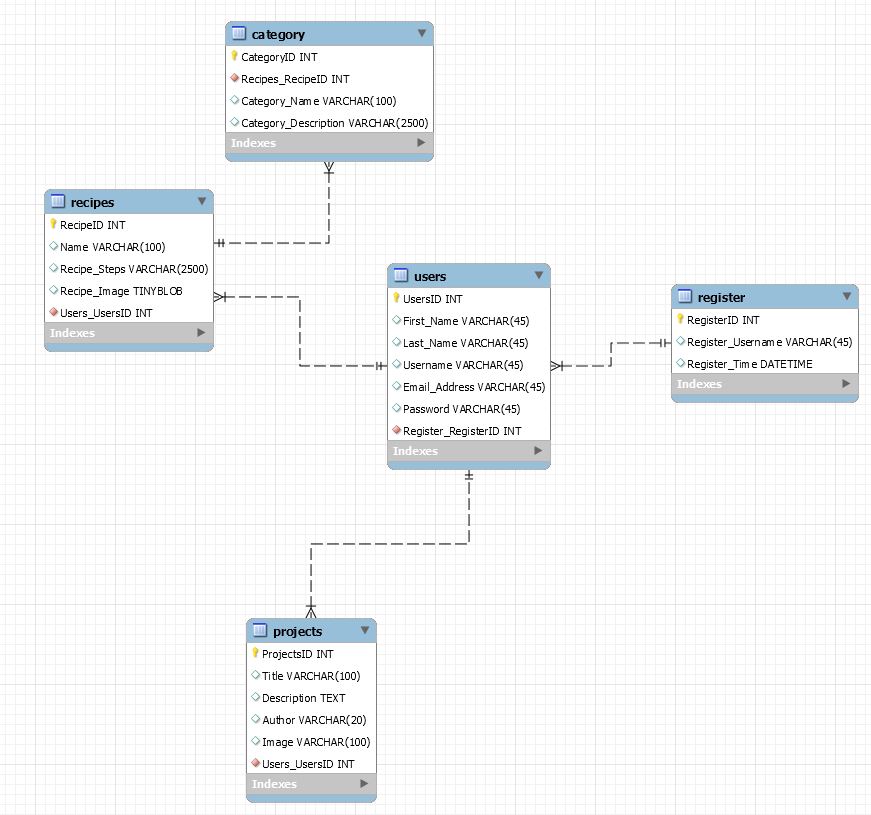
# Data Access Design

## Model Design (Data Model)

*This is the “model” part of the MVC pattern.*

*We refine the high level business domain model and create a detailed data model with all the entities that we will need to store in the database, with their attributes and relationships.*

High Level business domain model:



## Functional Decomposition

*Select the key entities in the data model. From the user stories identify the types of functions needed on each entity, and which users will be performing them. This will provide you with a basis for identifying the Views that you will need.*

Summary of User stories:

Functions needed:

1. View different Categories
2. Find Recipes related to different categories
3. Show images related to categories and recipes
4. Add/Delete/Modify categories and recipes
5. Add/Delete/Modify Users/Clients
6. Give permissions to authorized users

User Story One:

1. User/Clients “As a user of the Web App I would like to have access to a large variety of food categories, with various easy to make recipes linked to the category titles were new recipes are added frequently. To save me not having to spend money on expensive cookbooks”

User Story Two:

1. Users/Client “As a user of the Web App I would like to view various recipes that I can prepare for family and friends I would like the Web App to be easy to use.”

User Story Three:

1. Administration “As an Admin staff of the Web App I would like to have permission to access to all database entries, so I am able to complete the following. Add/Edit/Delete from all table of the web App database.

User Story Four:

1. Users/Client “As a user of the Web App I would like to be able to view a detailed description and some images, that relate to the cooking recipe I intend to use.”

User Story Five:

1. Staff “As a staff member I would like to be able to have certain permissions based on the work I need to complete.”

## Database Investigation

*List which database management systems you investigated, and which is more appropriate for your web app. Justify your decision.*

Database: We’ll being using SQlite3 which is bundled with Django which allows for quick installation and able to utilize the tables which within the application.

# Security Design

## Security Implementation

* Define security requirements: Identify and document security requirements early in the development life cycle and make sure that subsequent development artefacts are evaluated for compliance with those requirements. When security requirements are not defined, the security of the resulting system cannot be effectively evaluated.
* We will implement security steps as soon as we have setup our development environment by assigning username and passwords for access to development and roles and permissions to our development team.
* Validate input: Validate input from all untrusted data sources. Proper input validation can eliminate most software vulnerabilities. Be suspicious of most external data sources, including command line arguments, network interfaces, environmental variables, and user-controlled files.
* We will setup different test users with different roles and permissions for testing the application when appropriate.
* Keep it simple: Keep the design as simple as possible. The more complex the design becomes the more likelihood there is to be errors.
* Sanitize data: Such as command shells, databases and COTS components. Attackers can invoke unused functionality in these components using SQL injection attacks.
* Quality assurance: Identify and eliminate vulnerabilities. Fuzz testing, penetration testing and source code audits should all be incorporated as part of a good quality assurance program.
* We will make use of external reviewers during test phase.
* We will apply a secure coding standard for our team.
* Two Factor Authentication for all Users/Clients.
* Back-end Set Administrator Privileges and login passwords and Usernames.
* Set Registration for Users/Clients with regular up to date renew and reset password functions.
* Inform Users/Clients to regularly change their passwords if needed.

## User roles and Permissions

* Admin/SuperAdmin: Can do everything, work with entries, create and delete, updates content types and the database.

Read, create, edit, delete, Unpublish, Unarchive.

* Recipe Creator: Can work with and edit entries and content but cannot delete.

Read, create, edit.

* Guest: Can edit and update entries.

Read, edit.

* Guest: Only view the pages and not able to access anything expect for requesting a password reset.  
  Update-Request

## Iterations

* Iteration 1: Apply a secure coding standard for our team.
* Iteration 2: Security for development environment, User roles and permissions
* Iteration 3: Implementation and testing.

# User Interface Design

## Required Views:

* Client/user story: “I would like to find new dinner recipes quick and easily with little effort or time as I am very busy” – We need to make sure the application is responsive on any device a user has available. It will be developed for mobile, tablet and PC this applies to the small business owner’s story also.
* Administrator user story: “I would like to be able to have the correct permissions to access the database in order to complete my workload” – We will make sure the user interface for the database is using appropriate names and labels with a clean, easy to read interface for the administrators to carry out their work.
* Stakeholder user story: “I would like to be sure the application I’m investing in will have performance, security, availability, reliability, portability and usability for all users of the web app”. – We will make sure to have a modern and simple design with easy to read menus and scalability across the app. We will have a simple but eye-catching colour scheme running across the whole application.
* These requirements are the same for the other user stories we have acquired during research.

## Overall Style:

* Colour palette – Mostly black and white but with pops of colour taken from the logo for menu title names and the background colour of some divs.
* Logo colour – Black, yellow, blue, green.
* Font – Biome.
* Font size – 12, Titles 16.
* Refer to wireframes for the page layouts.

## Style Wireframe

Footer

Images here

Txt Here

Quick & Easy Recipes



P A Z