Name: Yandong Sun(ys2312), Yangkun Huang(yh1875)

**1. Abstract**

We design a website called cookzilla™ that focuses on cooking and recipes. This site allows people to post cooking recipes, to see recipes posted by other members, to review and grade posted cooking recipes, to attach additional suggestions to a posted recipe. In this website, people can see all the recipes, all the events and all the groups. It also allows people to create and join the informal groups and organize the events. The members of the group can RSVP for such meetings, and may post meeting reports with photos of the event to the site. When the people visit this websites, they can sign up to be the member of this cookzilla, of if they have already been a member, he or she can sign in to this website. Moreover, they can

**2. Introduction**

Here are some more details about the idea behind this new site.

Users can register on this website, it needs password username and email while the city of users live in, the birthday of the users, the description about themselves and the profile icon are optional. When a registration has done, a new record is inserted into the User table. User is identified by a unique uid. Users can post recipes on the site that are then visible by everybody. A recipe usually has a title (e.g., “Aunt Mary’s Apple Pie”), a number of servings, the ingredients and their quantities, a textual description of how to make the food item, and one or more pictures of the this food. User can search for particular recipes, groups, events and users.

When users sign in to website, they can see their profile and edit it. They can see all the recipes that they posted, they can choose delete it or not, and they can see the detail of the recipe by clicking “detail” button. They can see all the events that they created, they can choose cancel it or not and they can see the detail of the event by clicking “detail” button. They can see all the RSVPs they have done, they can choose to cancel it or not and they can see the detail of the RSVP event by clicking “detail” button. They can see all the groups that they have joined, they can choose to quit or not and they can see the detail description of the group by clicking “detail” button. They can see all the comments that they have created, they can choose to delete them or not and they can see the detail description of the recipe that the comment belongs to by clicking “detail” button. They can see all the reports that they have created, they can choose to delete them or not and they can see the detail description of the event that the report belongs to by clicking “detail” button.

By clicking the detail button of recipe, user can see the detail of the recipe which includes the photos of the recipes, the tags of the recipes, the links to other related recipes, and the title, serving number, description, post time, creator, ingredients, and comments of the recipe. At the bottom of this page, users can choose to add comments to this recipe, give it some suggestions on how to modify the recipe (e.g., “use butter instead of oil” or “bake at slightly lower temperature for a more moist cake”)., add some review photos to it and give it a rating from 1 to 5.

The system also has a set of predefined tags, supplied by the website, to help organize recipes into categories. Examples of such tags would be “italian’’, “chinese”, “vegan”, “soup”, “spicy”, etc. When a user posts a recipe, they can attach one or more of these tags to their recipe. As for the links to related recipes, here is how we define “related”: if two recipes have the at least one same recipe, we say that these two recipes are related.

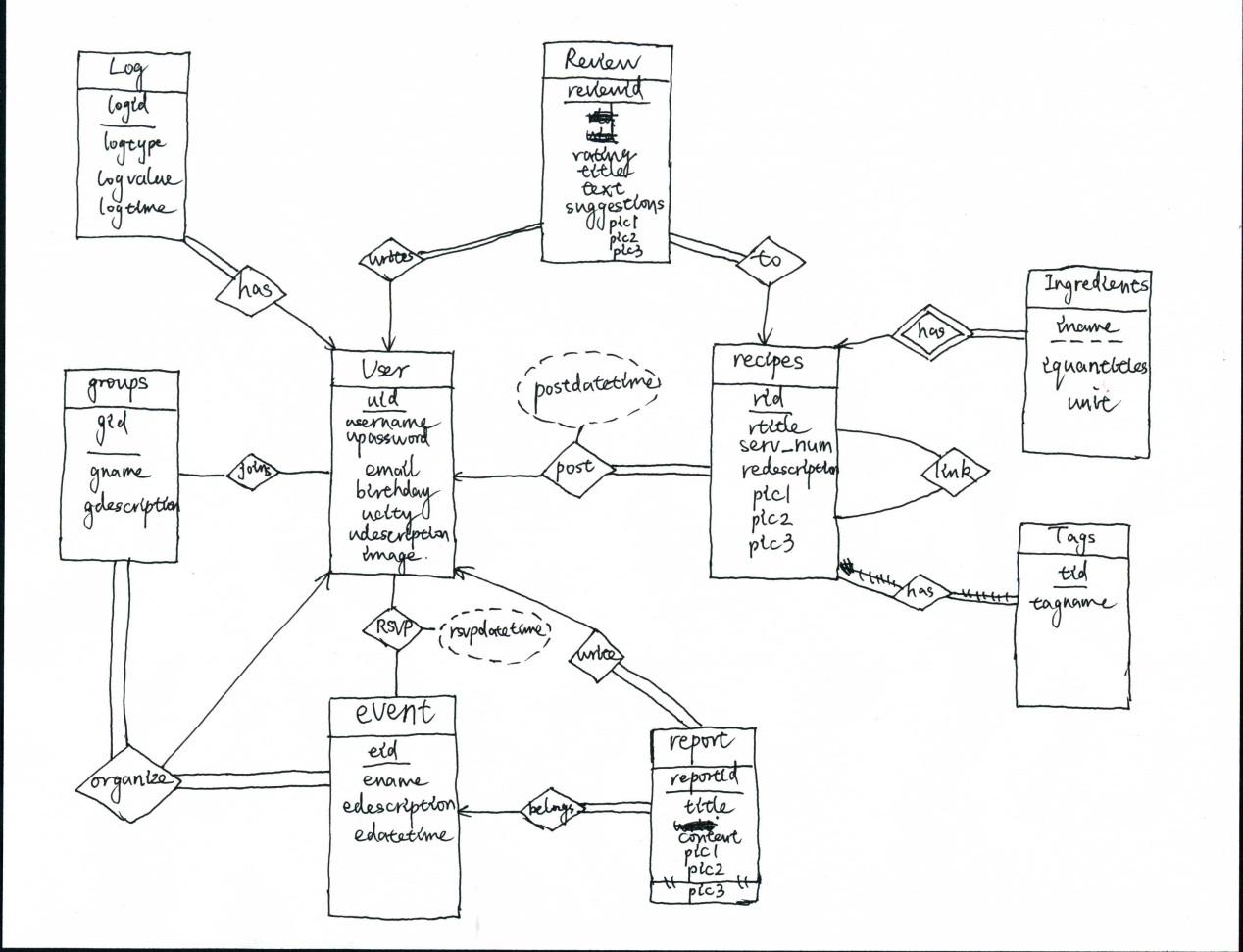
By clicking the detail button of event, users can see the detail description about the event which includes id, name of event, description about event, time it will start, creator of the event, and the group that this event belongs to. Also users can see all the attenders of this event, and all the reports about this event. At the bottom of this page, users can choose to add report to this event which includes a title of this report, a description about this report and three picture.

By clicking the detail button of group, users can see detailed description about the group which includes all the members of this group.

Further users can edit their profiles they can change the username, password, birthday, city, description of their profile.

On the top bar, users can see 4 buttons which are “Home”, “Recipe”, “Group”, “Event”. After logging in they can see “My History” button. “Home” is the home page of this websites. “Recipe” shows all the recipes. “Group” shows all the groups, “Event” shows all the events. After logging in they can see “My History” button, which shows all search history, all the tags that they clicked on before, all the recipes they have visited before. These records can help users understand their habit better. They can choose to delete these logs.

**2.1. ER diagrams**



**2.2.Tables**

User(uid, upassword, username, ubirdthday, ucity, uemail, udescription, image)

This table is to store user’s information.

The uid is the primary key.

The user can sign up with the unique email with the password.

Username is the display name which other users can see.

When user signs up, the password, username and email cannot be null, while the birthday of the user, the city where user live in, the description about the user and the image of the user can be optional.

Recipes(rid, uid, rtitle, serv\_num, rdescription, postdatetime, image)

This table is to store recipe’s information.

The “rid” is the primary key. It is auto increment. Every recipe should have its own unique rid.

The “uid” is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”. It shows that which user create this recipe.

The “rtitle” is the name of the recipe that user who create this recipe give.

The “serv\_num” is the number of serving of this recipe.

The “rdescipstion” is the description of this recipe. User can write the instructions and steps to make the food. User can make it as detailed as possible and the description can hold 2000 charaters.

The ”postdatetime” is the date and time that user post this recipe.

When a user create a recipe and submit it, all of these attribute cannot be null. The “rid” and “postdatetime” are created automatically.

The “image” is the picture of this recipe.

Ingredients(rid, iname, iquantities, unit)

This table is to store ingredient information. Because each recipe could have one or more ingredient, we need this table to store ingredient information.

The (rid, iname) is primary key

The rid is the foreign key referencing Recipes (rid) and it has constraints “on delete cascade, on update cascade”. It shows that this ingredient belong to which recipe.

The “iname” is the name of the ingredient.

The “iquantities” is the quantity of this ingredient in the recipe.

The “unit” is the unit of the ingredient, and it is enumeration, which has ‘gram’, ‘milliliter’ and ‘item’. The ‘item’ represent that a whole food item such as an apple, a banana or a mango. Before the quantities inserted into the database, their units have already been converted into ‘gram’, ‘milliliter’ or ‘item’. So we only store three units in the database.

All the attributes cannot be null.

Link(rid1, rid2)

This table is to store recipe link information. The link is from one recipe to other recipe. One recipe could link to one or more recipes, and one recipe could be linked by one or more recipes. Therefore we need this table.

The (rid1, rid2) is the primary key.

The rid1 is the foreign key referencing Recipes(rid) and it has constraint “on delete cascade, on update cascade”.

The rid2 is the foreign key referencing Recipes(rid) and it has constraint “on delete cascade, on update cascade”.

All the attributes cannot be null.

Tags(tid , tagname)

This table is to store all the tags.

The tid is the primary key and it is auto increment.

The “tagname” is the name of the tag.

All the attributes cannot be null.

HasTags(rid, tid)

This table is to store the relationship between recipe and tags. One recipe can have many tags and at the same time, many recipes can have some same tags.

The (rid, tid) is the primary key.

The rid is the foreign key referencing Recipes(rid) and it has constraint “on delete cascade, on update cascade”.

The tid is the foreign key referencing Tags (tid) and it has constraint “on delete cascade, on update cascade”.

All the attributes cannot be null.

Review(reviewid, rid, uid, rating, title, text, suggestions, pic1, pic2, pic3)

This table stores the review of users including which user write this review on which recipe, its rating that user gives, its title, text and suggestions.

The reviewid is the primary key. It is auto increment.

The rid is the foreign key referencing Recipes(rid) and it has constraint “on delete cascade, on update cascade”. It shows that this review is on which recipe.

The uid is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”. It shows that this review is written by which user.

The “rating” is the rate that user give to the recipe. It should be integer and should be >= 0 and <= 5.

The “title” is the title of the review.

The “text” is what user wants to say to the recipe. The user can write 1000 characters at most.

The “suggestion” is what user wants to suggest to this recipe. The user can write 1000 characters at most.

The “pic1, pic2, pic3” are picture of the review, they are optional and all other attributes cannot be null except “suggestion” and pics.

Groups(gid, uid, gname, gdescription)

This table is to store group information. We should know which user create this group so we need store uid of the user who create this group. Also a group should have its name and description.

The “gid” is the primary key.

The “uid” is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”.

The “gname” is the name of the group.

The “gdescription” is the description of the group. Only the user who creates this group can edit this part.

All the attributes cannot be null.

Join(uid, gid)

This table is to show which user has joined which group. One user could join one or more group, and group could have one or more user as members.

The (uid, gid) is the primary key.

The “uid” is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”.

The “gid” is the foreign key referencing Group(gid) and it has constraints “on delete cascade, on update cascade”.

All the attributes cannot be null.

event (eid, gid, ename, edescription, edatetime, creator\_id)

This table describes the information about the event, including which user in which group create this event, the name of the event and its description, date and time. An event is organized by a user who is a member of a group, and this event belongs to a group. So we need store the user’s uid(creator\_id) who organize this event and the gid of the group which this event belongs to.

The “eid” is the primary key. It is auto increment.

The “gid” is the foreign key referencing Group(gid) and it has constraints “on delete cascade, on update cascade”. It shows which group organizes this event.

The “creator\_id” is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”. It shows which user organizes this event. The user should in this group.

The “ename” is the name of the event.

The “edescription” is the description of this event.

The “edatetime” is the date and time that this event will start.

All the attributes cannot be null.

RSVP(uid, eid, rsvpdatetime)

This table describes information about which user RSVP to which event. One user can RSVP to one or more event, and an event can be RSVP by one or more users. So we need store user ‘s uid and corresponding event’s eid.

The (uid, eid) is the primary key.

The “uid” is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”. It shows that which user do the rsvp.

The “eid” is the foreign key referencing Groupevent(eid) and it has constraints “on delete cascade, on update cascade”. It shows that user has rsvp on which event.

The “rsvpdatetime” is the date and time of this rsvp happens.

All the attributes cannot be null.

Report(reportid, title, writerid, content, pic1, pic2, pic3)

This table records all the reports, the user who creates the report and the content of the report.

The “reportid” is the primary key. It is auto increment.

The “writerid” is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”. It shows that which user finishes this report.

The “content” is the content of the report.

The “pic1, pic2, pic3” are the pictures of this report.

All the attributes cannot be null except three pictures.

Reporttoevent(reportid, eid)

This table shows that which report corresponds to which event. One report can only report one event, so we need this table to store the corresponding relationship.

The primary key is (reportid, eid)

The “reportid” is the foreign key referencing Report(reportid) and it has constraints “on delete cascade, on update cascade”. It shows that which report it is.

The “eid” is the foreign key referencing Groupevent(eid) and it has constraints “on delete cascade, on update cascade”. It shows that which event it is.

All the attributes cannot be null.

Log(logid, uid, logtype, logvalue, logtime)

This table shows users search history, all the recipes that users visited before and all the tags users clicked on before.

The primary key is (logid)

The “uid” is the foreign key referencing User(uid) and it has constraints “on delete cascade, on update cascade”.

The “logtype” is enumeration enumeration which has three items “recipe”, ”search”, ”tag”.

When logtype is recipe, the logvalue is the rid of the recipe.

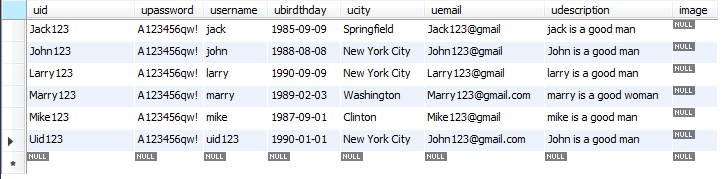
When logtype is search, the logvalue is the search keywords that user entered.

When logtype is tag, the logvalue is the tagname of the tag.

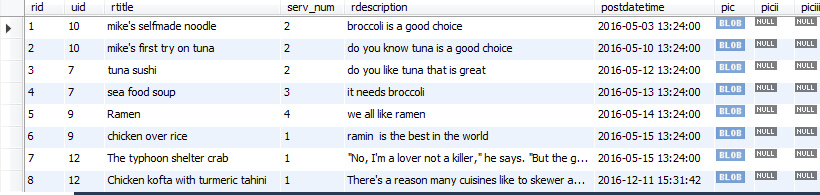
The “logtime” is the time that this log generated.

**2.3 sample data**

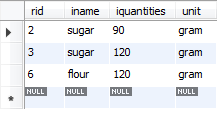
User



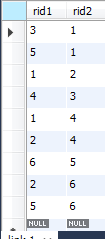
Recipe



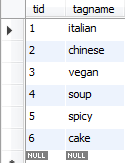
Ingredients



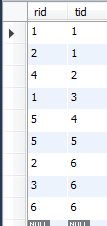
Link



Tags



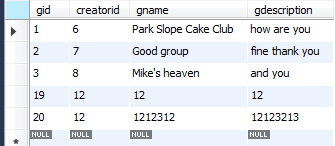
HasTags



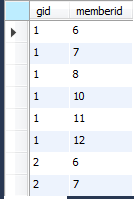
Review



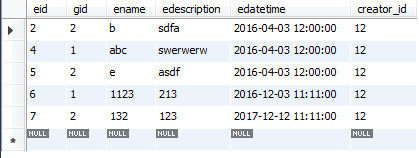
Groups



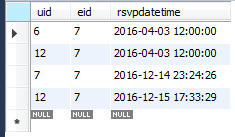
Join



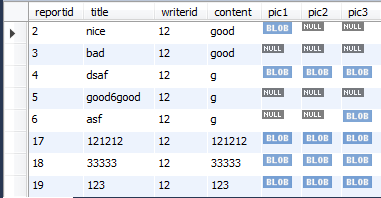
event



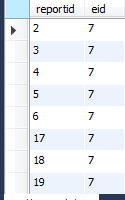
RSVP



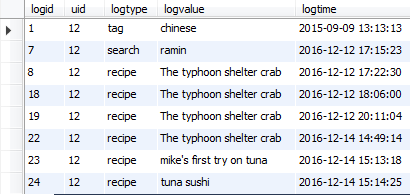
Report



Reporttoevent



Log

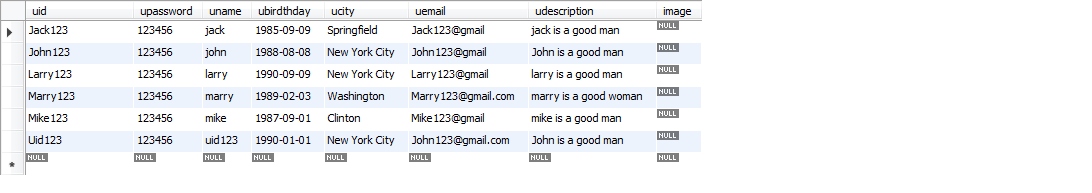


**2.4 Sample Queries**

1.Create a record for a new user account, with a name, a login name, and a password.

Insert into user values(‘Uid123’, ‘123456’, ‘uid123’, ‘1990-01-01’, ‘New York City’,  [‘John123@gmail.com](mailto:123@gmail.com)’, ‘John is a good man’, null);





2.List all recipes with tag “italian” that contain the keyword ``broccoli’’.

Select rid, rtitle

From recipes natural join hastags natural join tags

Where tagname= ‘italian’ and rdescription like ‘%broccoli%’  


3.List all members of the group “Park Slope Cake Club” that have given a positive RSVP to more than three events of the group.

Select rsvp.uid

From groups join groupevent join RSVP on groups.gid = groupevent.gid and groupevent.eid = rsvp.eid

Where gname= 'Park Slope Cake Club'

Group by uid

Having count(rsvp.eid) > 3



4. List all recipes with tag “cake” that contain more than 50 grams of sugar per serving.

Select rid, rtitle

From recipes natural join hastags natural join tags natural join ingredients

Where tagname = ‘cake’ and iname = ‘sugar’ and unit = ‘gram’ and iquantities/serv\_num > 50



5. Add a review with title “Yummy!”, text “Really, really, tasty!”, and a rating of 5 stars to the recipe for “Grandma’s Fettuccini Alfredo”.

Insert into review (`rid`, `uid`, `rating`, `title`, `text`, `suggestions`) values ((select rid from recipes where rtitle='Grandma’s Fettuccini Alfredo'), 'John123', 5, 'Yummy!', 'Really, really, tasty!', 'no one');





6. List all recipes containing the word “tuna”, sorted from highest to lowest average rating

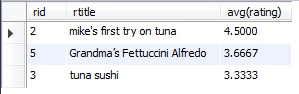
Select Recipes.rid, Recipes.rtitle, avg(rating)

From recipes join review on Recipes.rid = review.rid

Where rdescription like '%tuna%'

Group by Recipes.rid, Recipes.rtitle

Order by avg(rating) DESC



7. List all recipes that are related to a recipe that contains the word “tuna”.

Select rid1

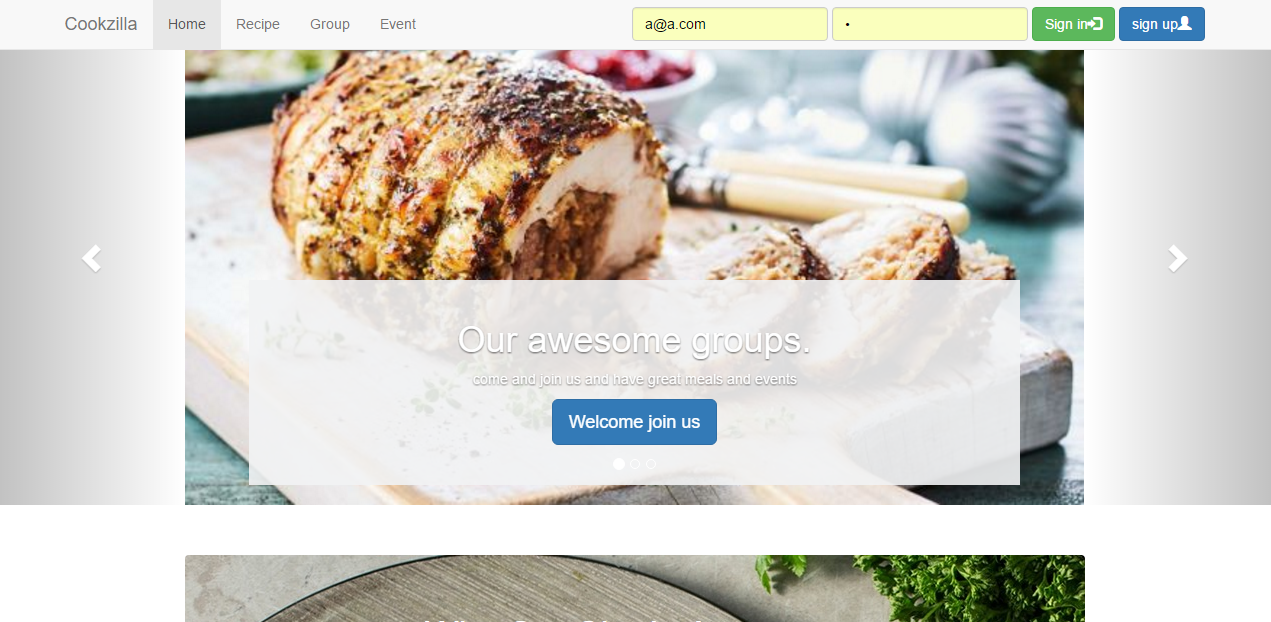
From recipes as r1 join link as l join recipes as r2 on r1.rid = l.rid1 and r2.rid = l.rid2

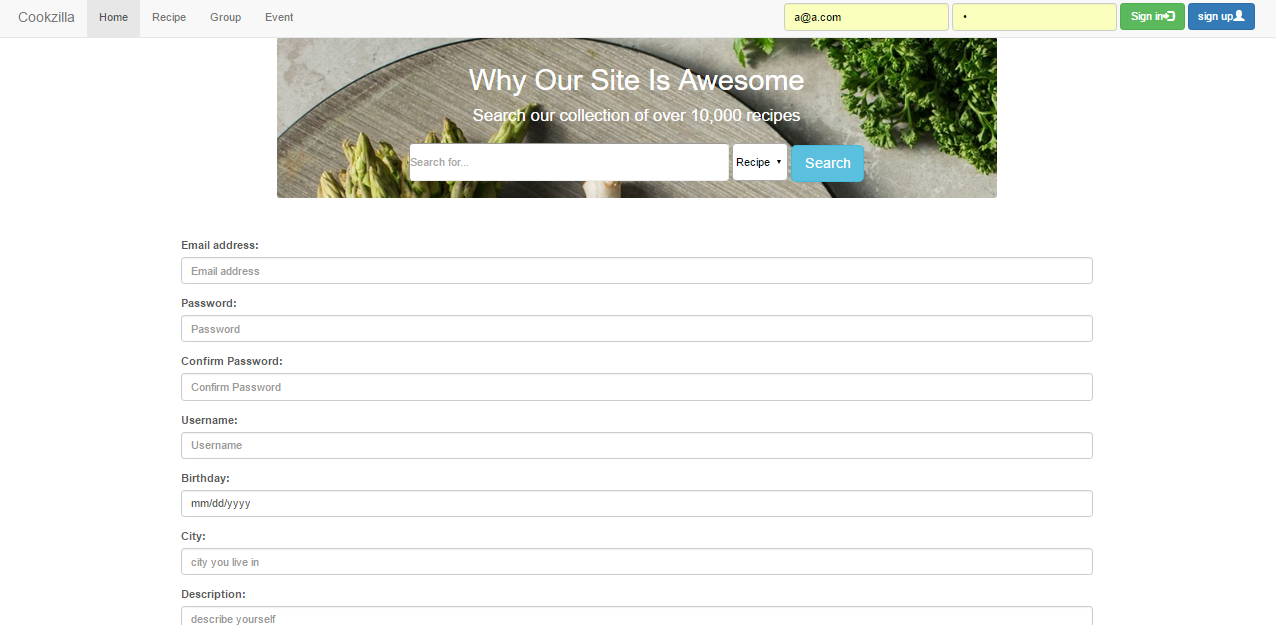
Where r2.rdescription like '%tuna%'



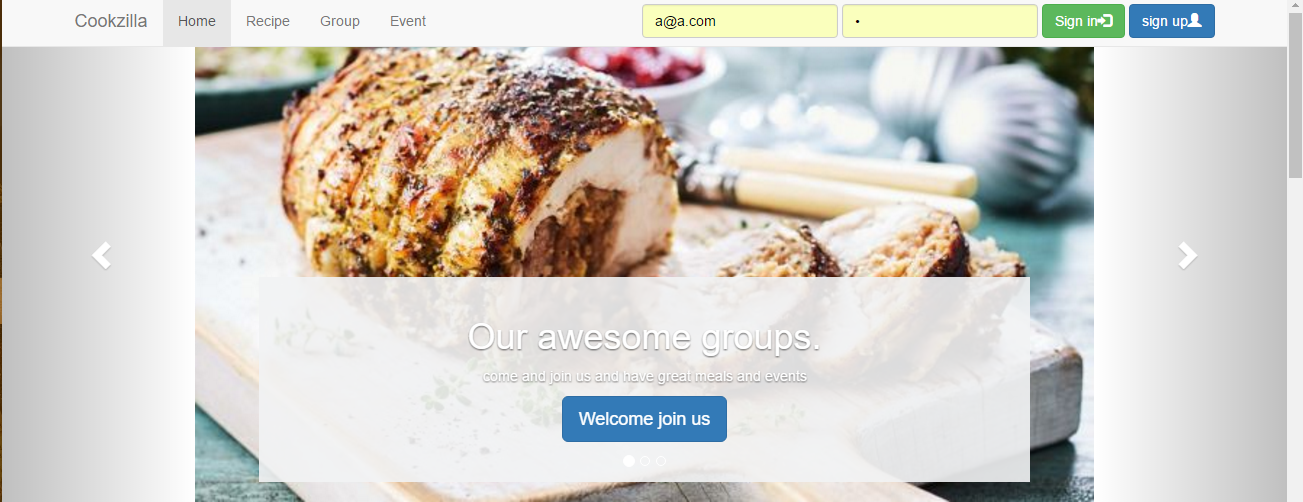
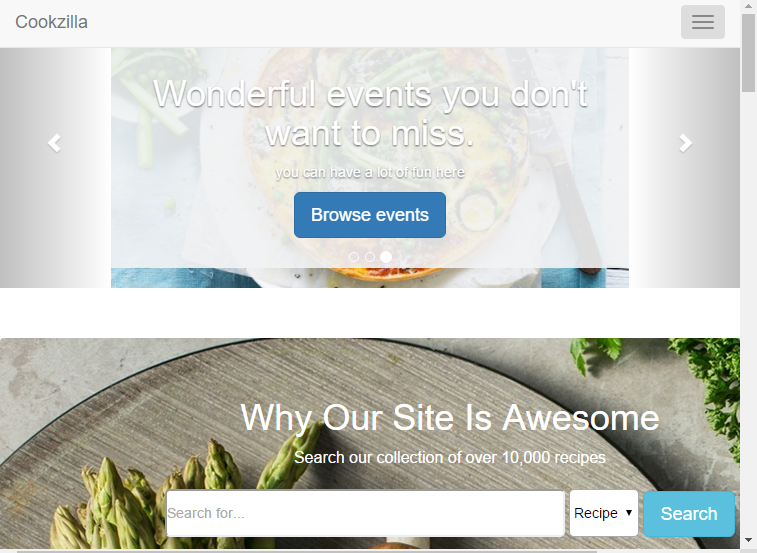
# **3.** Website Interface

1. First is the homepage and signup page

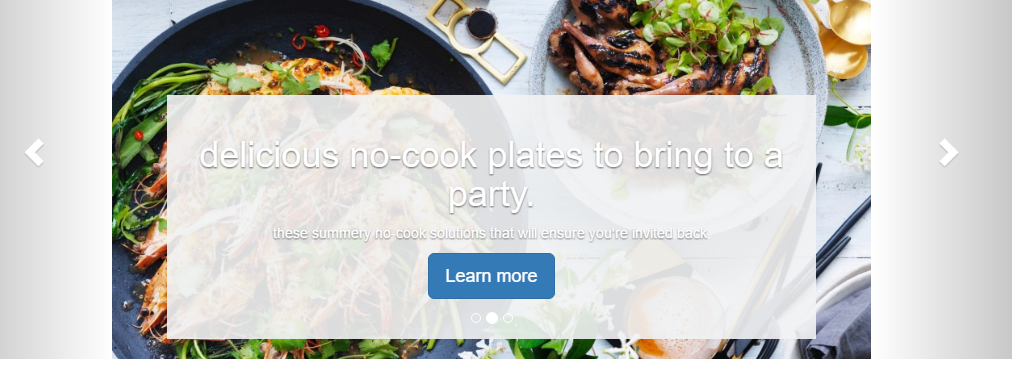




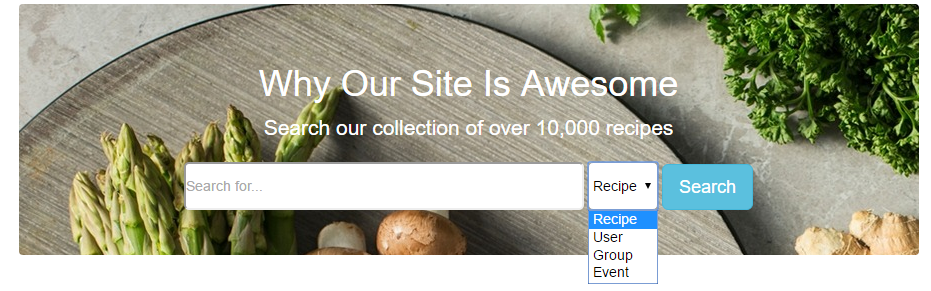
1. The navigation bar is different when window size changed.



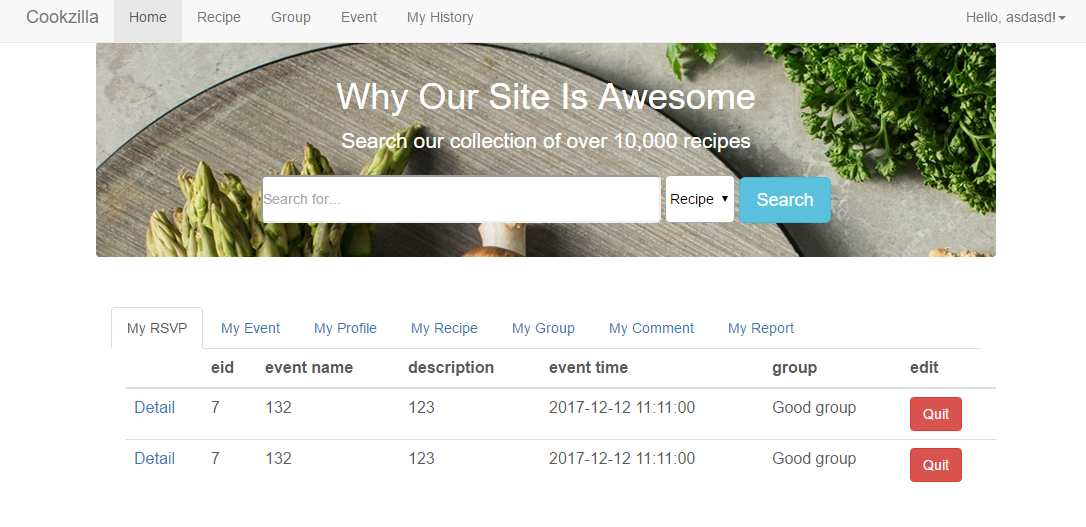
1. A automatically sliding Carousel



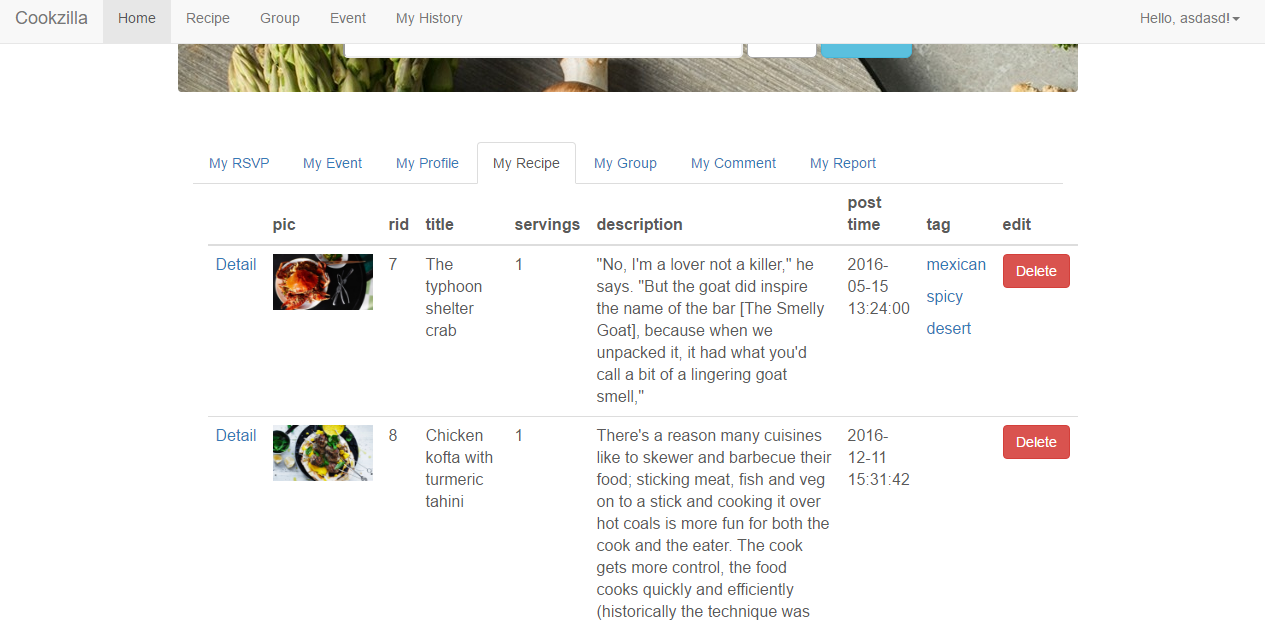
1. A search bar which could search by user, recipe, group and event.

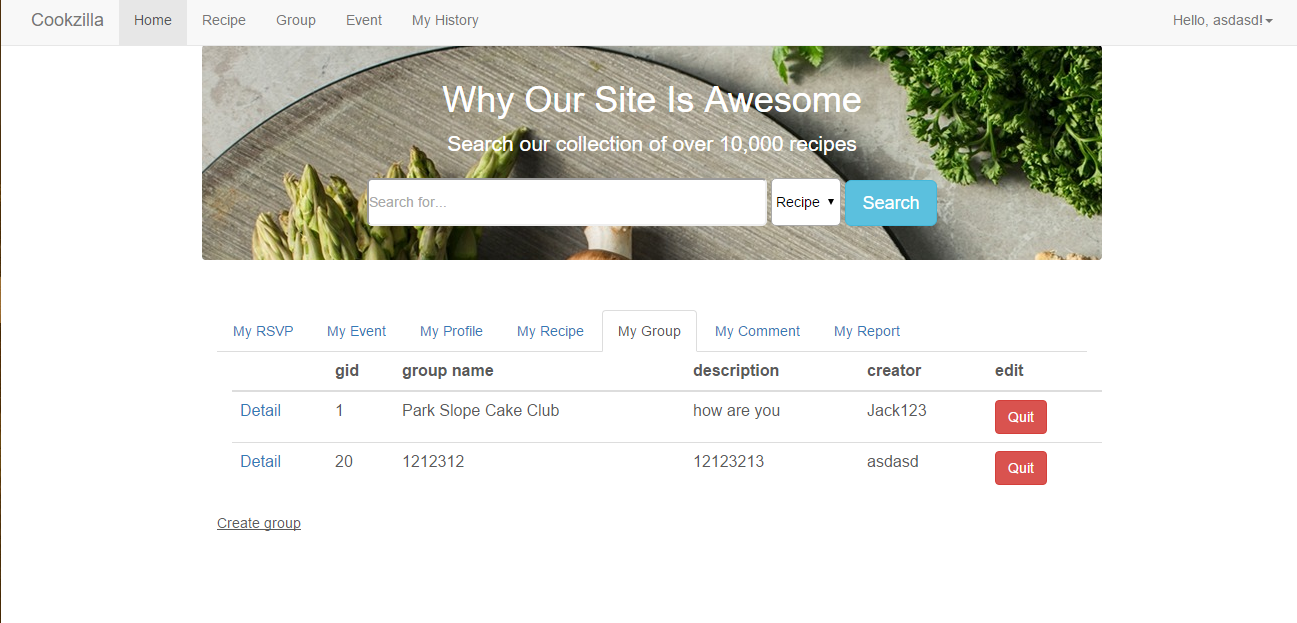


1. After login, enter the User’s home page. The navigation bar also change to display greeting words.



1. There are different divisions which display related information





1. After login, there is one more tab “History” in navigation bar which display user’s past search, browse, and tag activity history

