**Ideas for Mr. Kupido**

1. Recipe-related
   1. Directions
      1. Everything on one plate is one recipe (including side dish)
      2. Everything else is only dish recommendation (bread, pickles, wine, etc.)
      3. All recipe must be implemented as a method of a .net class
      4. All classes and methods referenced in the recipe’s code must be strictly from the predefined namespaces of the ingredients, tools, devices. Recipes can also be referred but their methods are not allowed to call.
      5. Users can select different versions of the recipe
      6. Users can replace ingredients easily
      7. If we implement a new version of a recipe, it should be done with copy-pasting the original, and then modifying the code
      8. Recipe as Flow-chart
      9. Heating up something should be measured in Joules as well
   2. Ingredients database
      1. All ingredients must be in a hierarchical tree structure, stored as .net classes
      2. All ingredients must have nutrition information in the database
      3. Users need to be able to set which ingredients they have or don’t have
      4. Ingredients should have latin-named classes
      5. Web page to display its name, latin name, wiki page, nutrition info, estimated retail price, etc.
   3. Nutrition database
      1. Nutrition information for the ingredients must be editable by users
      2. Nutrition information of recipes should be cached in database
   4. Price database
      1. Maintained by users as well
      2. Filled up by store-chain prices (Auchan, Tesco, Spar, Match, CBA, Cora, Coop, Aldi, Lidl, etc.)
      3. Localize prices, store location of the shop/price as well
      4. Look for seasonal price drops
      5. Include price and distance to the total price calculation
      6. Select between “cheap” or “convenient” shopping: convenient means shopping in only one shop, cheap means going around in the city to get the best prices
   5. Menu creation
      1. Create menus from recipes
      2. Create menus for the week
      3. Offer or order menus from users in your area
   6. Shopping list generation
      1. Generate shopping list for the day/week
      2. Order ingredients by category
2. Analytical engine
   1. Class library
      1. Every ingredient is a class
      2. Every recipe is a method in a class
      3. Every tool/device is a class
      4. Replacement attribute:
         1. Used for ingredients and tools
         2. Only the root class should be defined, all of its children should be acceptable
      5. InlineByDefault attribute
         1. Used for recipes
         2. If set the default action is to include the preparation directions into the recipe which refers to it (e.g. fried rice, pommes frites, etc.)
         3. If an ingredient is a recipe, we suppose that is was bought (unless it has the Inline attribute)
   2. Database
      1. Ingredients must have a connected database entry which contains its nutrition information, typical expire times, storage temperatures, default units (as a syntax sugar), density
      2. Recipes must have a connected database entry which contains the results of the analytical engine (e.g. preparation time, difficulty, costs, language specific information, serving method, serving temperature, etc.)
   3. Tags
      1. Tags are automatically generated, but they can be defined or deleted manually
      2. Auto-tags are methods defined in a class
      3. Auto-tag methods are executed once per recipe just after completion
      4. Diets are auto-tags to check if a recipe fits the diet (low-carb, paleo, vegan, etc.)
      5. Categories/types are tags (starter, main dish, side dish, dessert, beverage, soup, poultry, fish, lamb, pork, pasta, ice cream, cake, salad, pottage, snacks, sandwich, burger, pizza, vegetables, coffee, tea, wine, spirits, cocktail, soft drink, beer, breakfast, supper, dinner, Hungarian, Indian, German, etc.)
   4. Analytical engine generates
      1. Ingredient lists
      2. Similarity indexes,
      3. Recipe’s nutrition information
      4. Preparation, cooking, dishwashing times
      5. Costs
      6. Needed devices
      7. Workouts to complete to compensate calorie intake
   5. Analytical engine inputs
      1. How many people are working on the recipe
      2. Which ingredient should be handled as inline (to be made) or referred (bought)
3. Search engine
   1. Search-text field to enter ingredients, tools, devices, recipe names
   2. If an ingredient/tool/device/recipe/type (soup, main dish)/diet/”pasta”/”fish”/”less than 15 minutes”/”less than $100” is recognized then indicate that on the UI and do not try to use it as a free-text search
   3. Offer “IntelliSense”-like dropdown in the search-text field
   4. Free-text search is only available for recipe names
   5. Use + and – to indicate “must have” and “shouldn’t have” logic
   6. On-the-fly result number estimation
4. Localization engine
   1. Generate localized, humanly understandable directions
   2. Allow to change the measurement units
   3. Allow to use “syntactical sugars”, like replacing “5g of salt” to “1 teaspoon of salt”
   4. Localize, URLs, recipe names, ingredients, tools/devices
   5. Use different currencies, refresh rates in the background
5. Security
   1. Use OpenID for authentication
   2. Make donation possible with BitCoin
   3. Limit maximum queries per IP address
   4. Use cookies to track unregistered users
   5. Limit likes/dislikes per IP/cookie id per recipes
   6. Limit the web page request per user per 1 minute and 24 hours
6. User profile
   1. Use default profile settings using information from Google+, Facebook, OpenID profiles
   2. Like and dislike without registration
   3. Likes and dislikes expire
   4. Make it possible to add friends/guests to the profile
   5. If a friend/guest were referenced more than 5 times as participant of the meal, ask for more details, invite them to the site
   6. Set portion size multiplier per guest
   7. Set ingredient likes/dislikes for guests
   8. Set goals for guests (e.g. move to vegetarian/paleo/low-card diet, decrease trans.fat intake, lower weight/BMI, etc.) and monitor their progress
   9. Upload blood sample/medical data upload
   10. Condition definition (e.g. type 2 diabetes, high blood pressure, cancer, etc.)
   11. Recommend meals depending on condition and its success rate in the past
   12. Define and name (“home in NY, home in BP, at parents’”) the locations where you usually cook (to help shopping list generation, ingredients, tools database maintenance, food ordering option filtering, default guests, etc.)
   13. Define the tools/devices you have
   14. Define ingredient list you have (set specific quantity or set to infinity)
   15. Define ingredients you/others like or dislike, but keep in mind that it can change over time
   16. Newsletter subscription
   17. If he/she haven’t logged in the last 60 days, ask them why in an e-mail, tell them how much we improved since they last logged in
   18. Get accomplishment badges like “Chicken Angel”, “Healthy Goddess”, “Spicy Devil”, etc.
   19. Organize get-togethers, invite people and set roles (purveyor, cook, helper, guest, dishwasher)
   20. Set the level of skill as beginner,average,master and change the preparation times according to it
   21. As a goal, the user should be able to set up if he is a professional athlete/bodybuilder
7. URL Routing
   1. URLs must be language specific
   2. Root URL: [www.mrkupido.com](http://www.mrkupido.com)
   3. 3-letter ISO culture: ISO 639.2 codes of languages
   4. <root>/<3-letter ISO culture>: [www.mrkupido.com/eng](http://www.mrkupido.com/eng) or [www.mrkupido.com/hun](http://www.mrkupido.com/hun)
   5. Unique recipe name: replace spaces,language spec. characters, numbers
   6. <root>/eng/recipe/<unique recipe name> or <root>/hun/recept/<unique recipe name>
   7. <unique recipe name>/ingredients
   8. <unique recipe name>/nutrition
   9. <unique recipe name>/directions
   10. <unique recipe name>/flow
   11. <unique recipe name>/directions/<actor index>
   12. <unique recipe name>/flow/<actor index>
   13. <unique recipe name>/similar
   14. <unique recipe name>/ratings
   15. <unique recipe name>/serving
   16. <root>/eng/menu/<menu id number> or <root>/hun/menu/<menu id number>
   17. Menus have the same sub URLs as recipes
8. Hosting
   1. Discount ASP.NET $240/year
   2. HostForLife $113/year .NET 4 500MB/DB x2
   3. 1&1 $113/year .NET 3.5 200MB/DB x2
   4. M6 $119/year ∑50GB x50
9. Games
   1. Apply for Mr. Kupido “premium pack”, to have specially designed plates, glasses, tablecloth, apron
   2. Uploading photos for premium pack owners is possible
   3. The uploaded photos will be rated by the community
   4. Order Mr. Kupido food containers with 2-digit numbered top and bottom
   5. Which is more delicious for your eyes?
   6. Which has more sugar/calories/vitamin C?
   7. Spot the difference between the pictures!
10. WebAPI
    1. Access WebAPI by using application key/auth code combination
    2. Limit API usage, unlock the limit for paying users
    3. View lastly viewed/generated directions plan on mobile (set a specific URL to show a PNG/JPG workflow)
11. Bug reports, user feedback, error handling
    1. Feedback form on the master-page
    2. Regular polls about new features
    3. Feedback request after completing a meal:
       1. how well did you follow the instructions
       2. how much you/others liked it
       3. any recommendations for modification
12. Multimedia
    1. Music
    2. Fully recorded preparation process with a spotless kitchen. 3-4 full HD cameras recoding the session, then compose a video wide angle view by combining their streams
13. Business
    1. If store-chains want to be included in the price calculation, they have to pay for us
    2. Ingredient factories can pay for the recommendation of their goods in the shopping list
    3. To recommend a wine/dessert for a dish, they can pay
    4. The user could pay for hiding all the advertising
14. UI-related
    1. Tip of the day on the landing page
    2. Some of the tags are filter tags, they should be set in the UI, and show them in the search results
    3. Fonts to use: Tangerine (type: regular, bold; size: 44,26) and Cardo (type: regular; size: 22,14)
    4. Basic colors to use: #513A2D #785643 #9E7259 #ADADAD #525252

2012.06.24. 14:15-19:15, 5 hours, Attila, first design attempts for master page and logo  
2012.06.26. 18:30-22:00, 3.5 hours, Attila, landing page refinements  
2012.06.27. 19:00-22:00, 3 hours, Attila, same as before  
2012.06.29. 19:00-22:00, 3 hours, Attila, same as before  
2012.07.01. 18:30-22:00, 3.5 hours, Attila, same as before (v10)  
2012.07.02. 18:30-21:30, 3 hours, Atttila (v12)  
2012.07.03. 18:30-22:00, 3.5 hours  
2012.07.04. 19:00-20:30, 1.5 hours  
2012.07.08. 18:30-21:00, 2.5 hours