**Design Document**

The Flabbergasters - Aaron Fine, Abraham Gunther, Matthew Lister, Mike Andrade

Purpose Statement

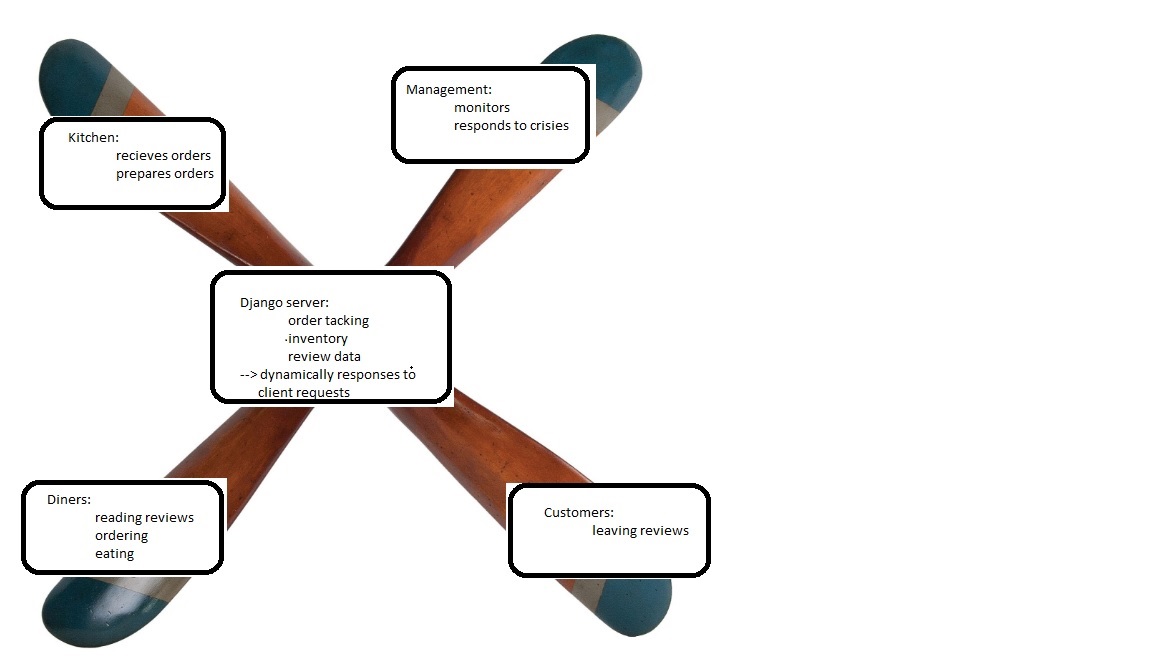
Running a restaurant involves keeping track of lots of little details and keeping many people informed about ever changing customer requests. Our software package will help manage the complexity of running a restaurant by taking care of order management, order communication, and order tracking, while providing management real time insights into their workflow. One key element of differentiation is that we plan to incorporate customer feedback into the managerial views as wells as for customers while they decide what to order.

Generally, we are trying to solve the following issues: customer satisfaction, inventory control, optimize staffing, order entry, order tracking, restaurant startup cost reduction, electronic ticketing system.

Design Outline

The Robot Waitress project decided to use a client server model. We are using on off-site server to house our database, system logic, and web page templates. The server will be a Debian Linux OS with python3 and the Django module using a SQLite database. We will be administering to the server through SSH. The server will then dynamically generate web forms for the clients based on the view/role the client requests. The clients will be using standard web-browsers (optimized for chrome but compatible with all major browsers). This allows the system to be hardware agnostic towards the clients. The clients do not require special software installed, and our program does not require elevated privileges on the client machines. We believe this is much more in keeping with the BYOD movement seen in workplaces today.

In a restaurant, orders are passed back and forth through several stages before the customer finally receives their food. The robot waitress project attempts to streamline the workflow by simplifying the production path and automating record keeping. Logically the system will form a propeller shape as orders are passed from the customers to the server, from the server to the kitchen, and back. Combined with a management interface to resolve issues, and satisfied customers sending feedback to the database, the robot waitress system actually goes through more steps than a traditional restaurant, but these are performed at wire speed and abstracted away from the users. The result being a simplified and more organized work environment.



Design Issues

**Functional Issues**

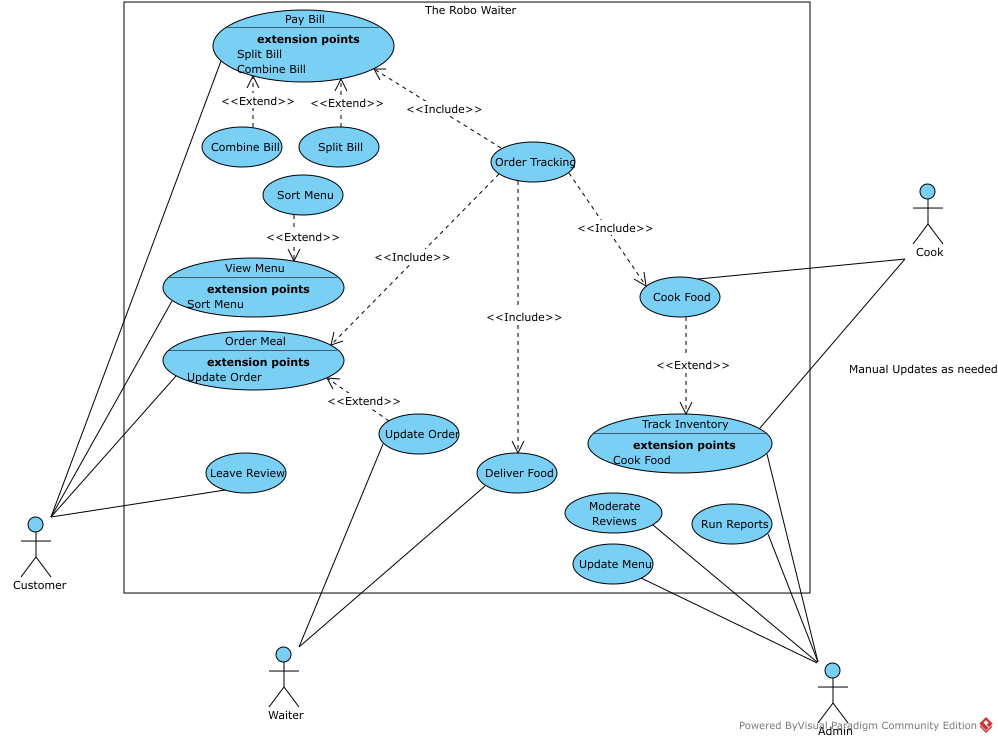
1. Customer interface functionality and aesthetics - Option 1, to use a web browser; Option 2, to use a mobile application - We chose to use a web browser as our customer interface since it easily integrates with a django server. It will also be accessible by mobile devices. The functionality will be managed by HTML, CSS, and JavaScript which gives us flexibility for design aesthetics.
2. Interface design - Option 1, Standard CSS template; Option 2, Customized interface design for each page - A standard CSS template will provide a familiar user experience as the roles of a user may change. Having a standard interface reduces the learning curve and training required to use the software.
3. Admin interface functionality - Option 1, Role-based functionality; Option 2, All in one screen - The decision is to have role-based functionality. This will make it easier to manage user accounts and allows distinction for a user’s interface based on their position in the company.

**Non-Functional Issues**

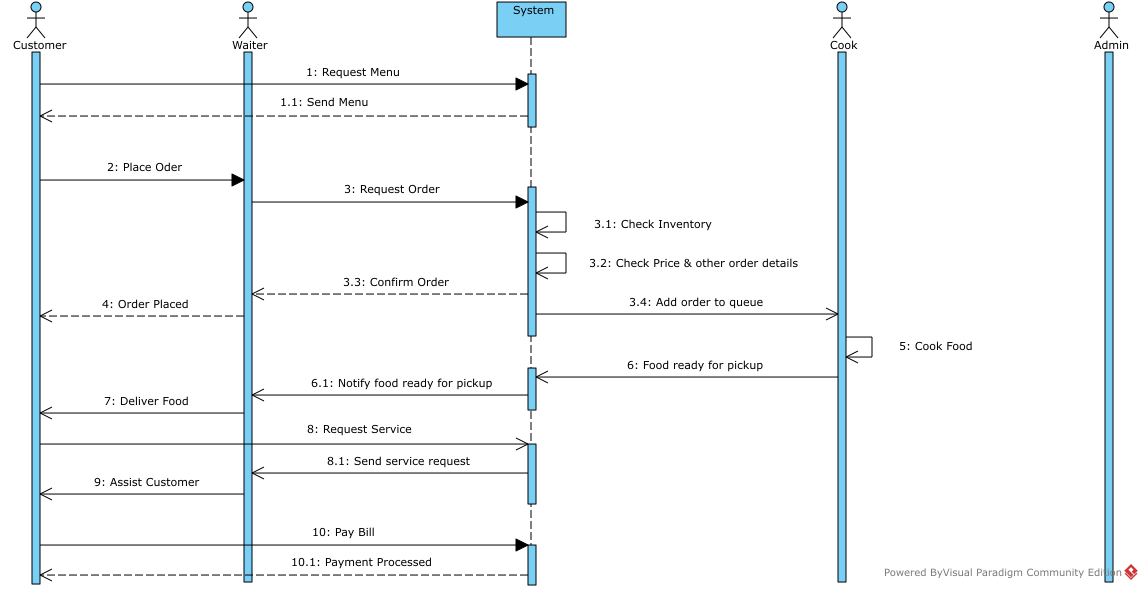
1. Database for housing critical data - Option 1, Django module using a SQLite; Option 2, MySQL - Our choice is to use Django since it uses python, which is our preferred language. Django’s use of tables will maintain a clear association between critical data for ordering and inventory control.
2. Communication between appropriate interfaces (and the server) - Option 1, Hub and Spoke; Option 2, Interface to interface interaction - We decided to use a hub and spoke interaction between the server and each of the interfaces. It allows the server to act as a gateway in order to verify any request.

Design Details:

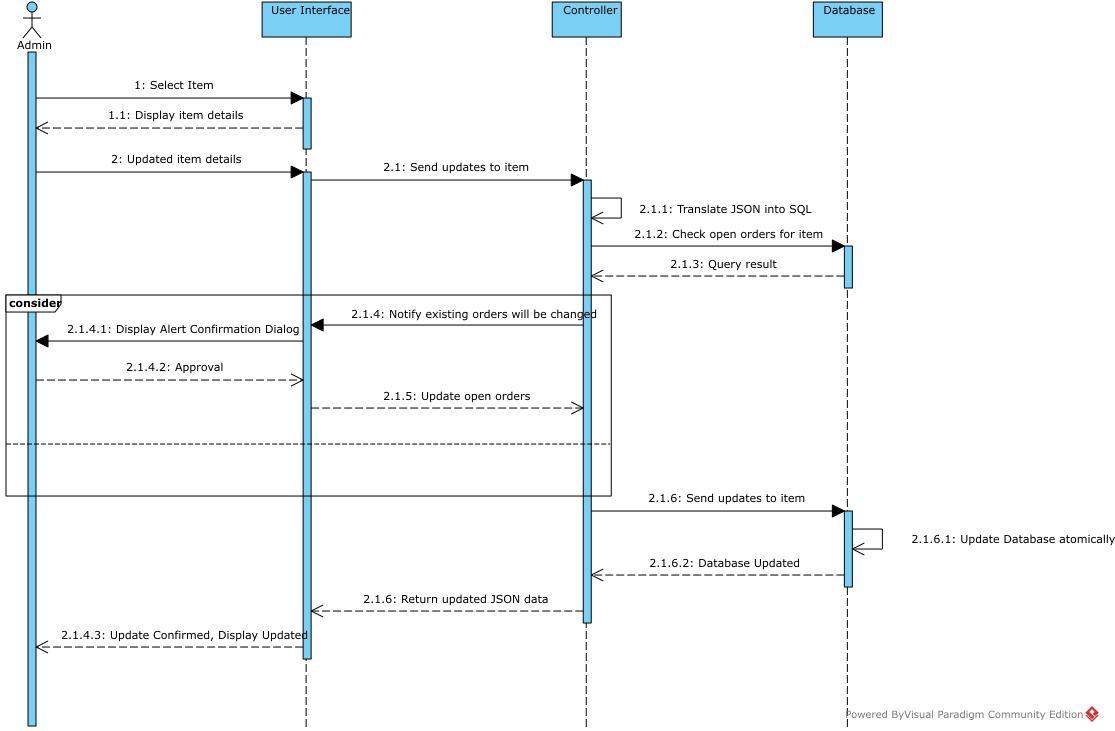
Use Case Diagram:



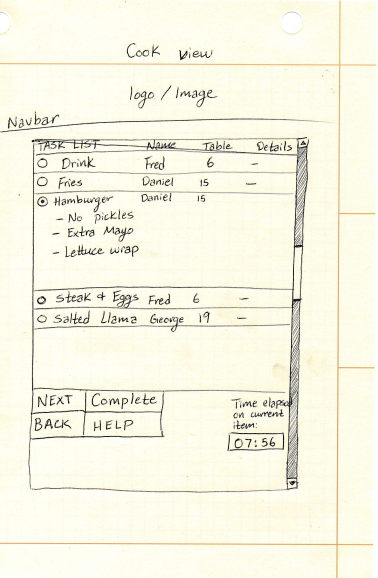
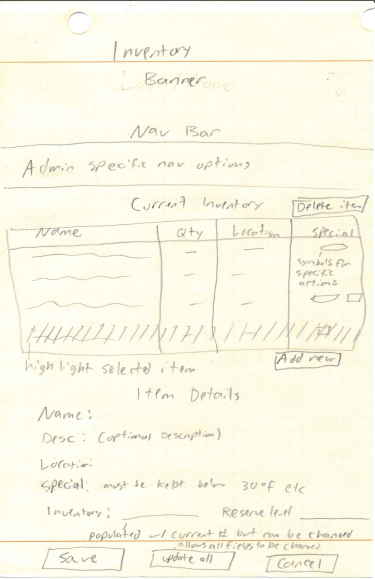
Basic Order Flow

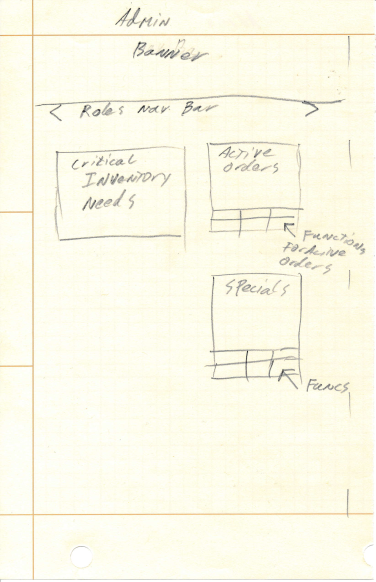
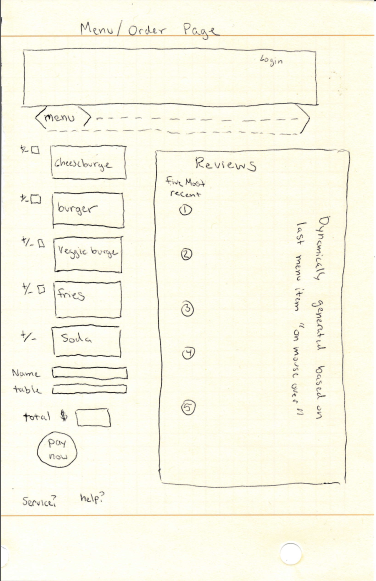


Updating a menu item



User Interface Design





Class/Database Diagram: