Ari Munkthur, James Gamarra, Jesse Ramirez

**Initial Test Plan:**

We plan implementing three modes of testing. First, is the single component test. This test will consist of breaking our application into each of its individual components in order to examine singular efficiency. This test allows our group to troubleshoot our application more effectively, thus allowing our group to single out each part that has problems instead of a holistic approach.

Second, our group plans on gathering a group of three individual volunteers to engage in usability testing. This will consist of the volunteer being instructed to navigate to our website and attempt to use the features. Our goal is to observe and take note of what the initial intended purpose of our website is and further gather information on how easy it is for users to achieve their goal through using our application. This will allow our group to make necessary design changing and observe possible additions in a non-biased atmosphere.

Third, our group will implement a python scripting automated testing feature that will populate our database with test information that we will then check for correctness of the database. This will be done using true-false, it will return a list that contains feature names and status.

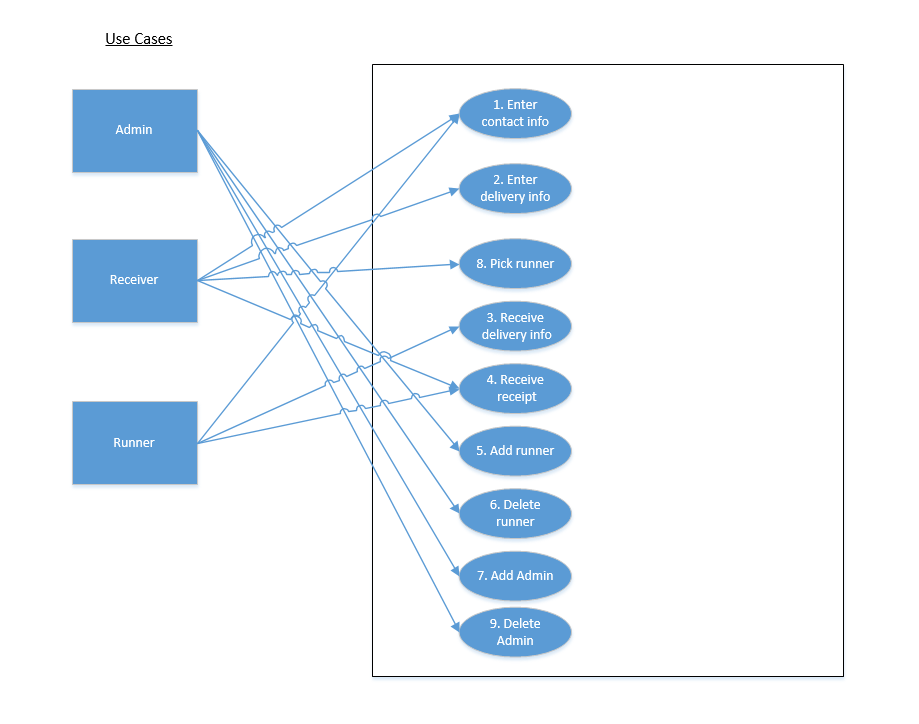
**Product Backlog**

|  |  |
| --- | --- |
| Python Test Script | 3 hours |
| Order HTML Page | 2 hours |
| Deliver HTML Page | 2 hours |
| Menu HTML Page | 2 hours |
| Receipt HTML Page | 4 hours |
| Active Busy Status Feature | 4 hours |
| Active Users List | 4 hours |
| Past Orders Feature | 4 hours |
| Past Order HTML Page | 2 hours |
| **Total** | **29 hours** |

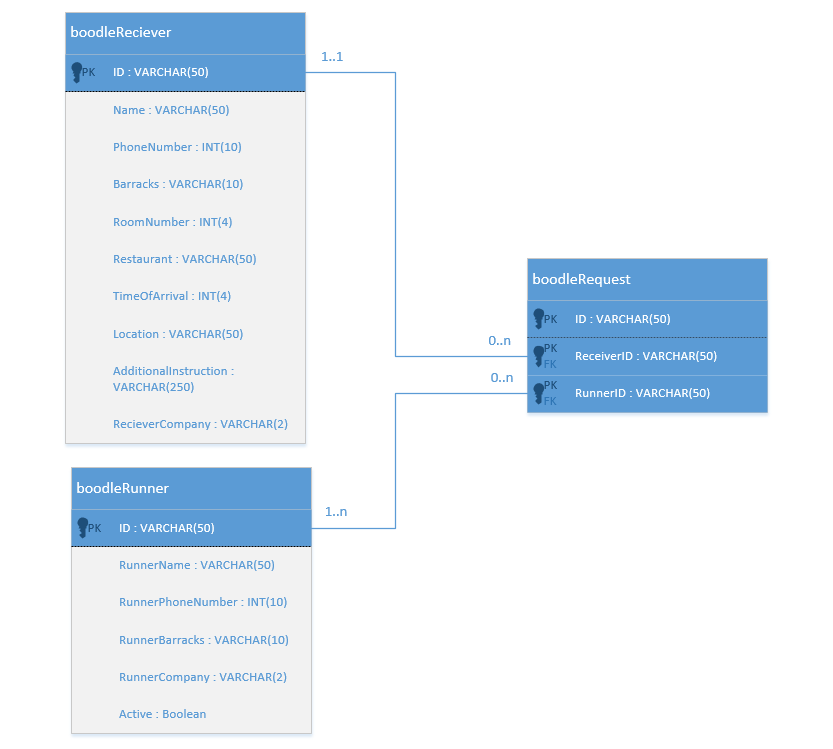
**Code Management Plan**

Throughout this project our team will use the fork and pull technique in GITHUB to manage our version history. This will allow us to clearly distinguish between implementation versions while also allowing us to push features to the current fork while we are working on it. In order to maximize efficiency we will maintain a GroupMe in which we will notify the other members of the group when we are working on a specific feature in order to reduce redundant coding.

**Use cases**



**Class Diagram**



Models.py

|  |
| --- |
| From django.db import models |
|  |  |
|  | class boodleReceiver(models.Model): |
|  | name = models.CharField(max\_length=50) |
|  | phoneNumber = models.IntegerField() |
|  | barracks = models.CharField(max\_length=10) |
|  | roomNumber = models.IntegerField() |
|  | restaurant = models.CharField(max\_length=50) |
|  | timeOfArrival = models.IntegerField() |
|  | additionalInstruction = models.CharField(max\_length=250) |
|  | receiverCompany = models.CharField(max\_length=2) |
|  |  |
|  | class boodleRunner(models.Model): |
|  | runnerName = models.CharField(max\_length=50) |
|  | runnerPhoneNumber = models.IntegerField() |
|  | runnerBarracks = models.CharField(max\_length=10) |
|  | runnerCompany = models.CharField(max\_length=2) |
|  |  |
|  |  |
|  | class boodleRequest(models.Model): |
|  | id = models.CharField(max\_length=50,primary\_key=True) |
|  | receiverID = models.CharField(max\_length=50) |
|  | runnerID = models.CharField(max\_length=50) |