**Development, Operation, and Maintenance Environments**

Android Burgerator will run as a native Android application on Android devices. Development, operation, and maintenance will utilize physical Android phones as well as virtual Android emulators.

TODO: Min Android API?

TODO: Target Android API?

**System Model**

* + High-level view showing the major components of the existing and proposed system:
  + Existing System:

IOS Burgerator (fig 1)



* IOS Burgerator is what currently exists. It is a mobile application that allows users to rate burgers within a geographic location (predominantly used in New York). Android Burgerator Base is intended to be an exact replication of IOS Burgerator.
  + Proposed Systems:

Android Burgerator Goal

Android Burgerator Base

* The proposed system Android Burgerator Base is ideally identical to IOS Burgerator. However, the Android Burgerator Goal is intended to be a more flexible extension of Burgerator that incorporates more social media aspects into Burgerator. For example, allowing you to share burgers with friends.
* TODO: Include a system level diagram for Android Burgerator.

**User Interaction**.

* + The user of the mobile application Andorid Burgerator Base, which is currently the focus for the project, is to allow individuals to visit a restaurant, take photos of a hamburger that they ordered and leave a rating for the burger in question.
  + Use-case diagrams and scenarios are an effective way to describe the interaction.
  + Refer to the use case diagram and corresponding scenarios(figs x-x)

**Functional Requirements**

* + The application must:
  + Allow the user to sign in(email, facebook, twitter)
  + Allow the user to review five main sections of the application(Find a burger,Burger Feed, burger rating, top burgers, user profile)
  + Allow the user to logout
  + Allow the user to control setting such as location from within the application

**Nonfunctional Requirements**. Detail the constraints under which your system must operate.

* + Given that Burgerator is location based, there must be access to location or a manual way to enter the location.
  + Constraints that the hardware imposes on the application are the same that other applications have. Memory, data, and battery constraints should be minimal.
  + The portability of the project is apparent given the underlying android Platform. This advantage opens up to application to the majority of the mobile maketshare.
  + The reliability of the application will rely on the servers that support it.

**Feasibility**.

* + Android base base,
    1. Base System Diagram(fix x)



* + android base,
    1. System diagram(fig x)
  + android goal base,
    1. TBD
  + android goal
    1. TBD

**Appendices.**

* + System Diagrams, ER or Database diagrams, Use-case diagrams, others as appropriate

**Use Case Diagram**



**Use Case Scenarios**

|  |  |
| --- | --- |
| Use Case Name | Login to Burgerator |
| Actors | User |
| Summary | The user logs into the application upon first use |
| Pre-Conditions | 1. User has the application installed  2. Internet connection is available  3. User has an Facebook account, Twitter account, or email to login with |
| Normal Flow of Elements | 1. User opens the Burgerator application  2. User is taken to the Burgerator splash screen  3. User is prompted login info  4. User chooses login account  5. User is logged into Burgerator |
| Error Conditions | 4a. User enters incorrect credentials  4b. User forgets account password |
| Concurrent Activities | 1a. Location is ascertained |
| Post-Conditions | 1. User is logged into Burgerator |

|  |  |
| --- | --- |
| Use Case Name | Find a burger/restaurant |
| Actors | User |
| Summary | Once logged into Burgerator, the user is in search of a burger/restaurant |
| Pre-Conditions | 1. User has the application installed  2. Internet connection is available  3. User location is enabled  4. User is logged in |
| Normal Flow of Elements | 1. User opens the Burgerator application  2. User is taken to the Burgerator home screen  3. User navigates to the ‘find a burger’ tab  4. User sorts by keyword, distance, or rating  5. User browses restaurants  6. User chooses restaurant  7. User chooses burger  8. User goes to restaurant |
| Error Conditions | 4a. No results returned  8a. Restaurant closed or burger no longer served |
| Concurrent Activities | None |
| Post-Conditions | 1. User has found a desired burger |

|  |  |
| --- | --- |
| Use Case Name | Browse burger feed |
| Actors | User |
| Summary | Once logged into Burgerator, the user browses the burger feed |
| Pre-Conditions | 1. User has the application installed  2. Internet connection is available  4. User is logged in |
| Normal Flow of Elements | 1. User opens the Burgerator application  2. User is taken to the Burgerator home screen  3. User navigates to the ‘burger feed’ tab  4. User browses other reviews  5. For every review, the user can:  View that review  View the review’s respective restaurants  View the review’s picture  ‘Pound’ the review  6. User continues to browse the burger feed |
| Error Conditions | 3a. Burger feed does not load |
| Concurrent Activities | None |
| Post-Conditions | 1. User has viewed rated burgers |

|  |  |
| --- | --- |
| Use Case Name | Rate a burger/ Add review |
| Actors | User |
| Summary | Once logged into Burgerator, the user attempts to review a burger |
| Pre-Conditions | 1. User has the application installed  2. Internet connection is available  3. User location is enabled  4. User camera is functional  4. User is logged in |
| Normal Flow of Elements | 1. User opens the Burgerator application  2. User is taken to the Burgerator home screen  3. User navigates to the ‘review’ tab  4. User chooses restaurant  5. User takes a picture of the burger  6. User rates the burger  7. User adds comments  8. User can share on Facebook and twitter  9. User submits rating |
| Error Conditions | 4a. User cannot find restaurant |
| Concurrent Activities | Content may or may not be posted to Facebook and twitter |
| Post-Conditions | 1. User has rated a burger |

|  |  |
| --- | --- |
| Use Case Name | Browse burger leaderboards |
| Actors | User |
| Summary | Once logged into Burgerator, the user browses the burger leaderboards |
| Pre-Conditions | 1. User has the application installed  2. Internet connection is available  4. User is logged in |
| Normal Flow of Elements | 1. User opens the Burgerator application  2. User is taken to the Burgerator home screen  3. User navigates to the ‘top 10 burgers’ tab  4. User browses top burgers  5. For every top burger, the user can:  View the top burger reviews  View the review’s respective restaurants  View the review’s picture  6. User continues to browse the burger feed |
| Error Conditions | 3a. Burger feed does not load |
| Concurrent Activities | None |
| Post-Conditions | 1. User has viewed top burgers |

|  |  |
| --- | --- |
| Use Case Name | Browse personal profile |
| Actors | User |
| Summary | Once logged into Burgerator, the user browses their profile |
| Pre-Conditions | 1. User has the application installed  2. Internet connection is available  4. User is logged in |
| Normal Flow of Elements | 1. User opens the Burgerator application  2. User is taken to the Burgerator home screen  3. User navigates to the ‘profile’ tab  4. User views their Burgerator rank (Squire etc…)  5. User browses previously rated burgers  6. For every top burger, the user can:  View the top burger reviews  View the review’s respective restaurants  View the review’s picture |
| Error Conditions | None |
| Concurrent Activities | None |
| Post-Conditions | 1. User has viewed their profile |

|  |  |
| --- | --- |
| Use Case Name | Manage database |
| Actors | Database Administrator (DBA) |
| Summary | The database administrators role is to clean garbage inputs from the system, modify the relational schema, and otherwise maintain the database |
| Pre-Conditions | 1. The DBA has access to the database  2. The DBA knows how to access the database |
| Normal Flow of Elements | 1. The DBA has the ability to:  Insert Inputs  Remove inputs  Modify the schema |
| Error Conditions | 1a. Database is unavailable due to hosting problems |
| Concurrent Activities | None |
| Post-Conditions | 1. The database has been maintained |

|  |  |
| --- | --- |
| Use Case Name | Maintain/Modify Android Application |
| Actors | Developer |
| Summary | The developer’s role is to create and maintain the application. |
| Pre-Conditions | None |
| Normal Flow of Elements | 1. The developers have the ability to:  Modify user interface  Modify database connection  Modify yelp api connection |
| Error Conditions | None |
| Concurrent Activities | None |
| Post-Conditions | 1. The application has been maintained |