**CMPS 5153 Software Engineering**

**Monday, September 12, 2016**

**Group: Cloud-MSU**

**Scott Gordon, Taylor Kirk, Vishnu Mandalapu, Tejaswi Singam, Mounika Mannam**

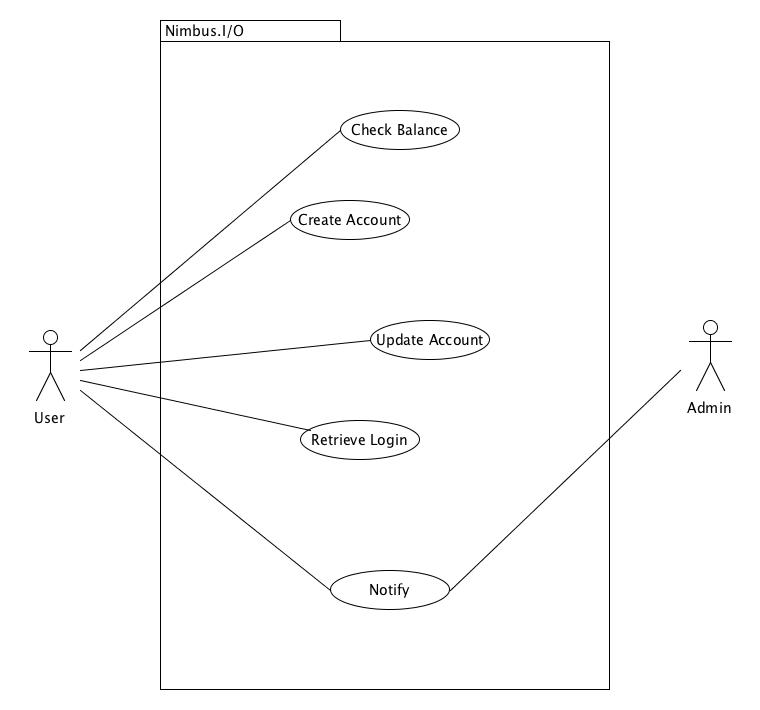
**Intent:**

*We have decided to implement a mobile order processing system designed for cloud services. This system will be known as* ***Nimbus.I/O.*** *The system will run natively on the most current android build (Currently, Android 7.0 “Nougat.”) Since the system will be built on Android, our programming language of choice is Java. For our development and testing purposes we will use Android Studio, which is the official Android integrated development environment, or IDE, built on Android’s Software development kit, or SDK. Along with Android Studio, we will use Github for source code management as well as WhatsApp for team communication and coordination.*

*Using our platform, customers will be able to select a wide configuration of hardware and software options. These selections correspond with various cloud-based solutions of hardware infrastructure, or IAAS (Infrastructure as a service) or software platforms, or PAAS (Platforms as a service), which are available for purchase from cloud based service providers. Take for example, a typical use case, known as ‘configure account’, where a user after creating an account, would log in and be able to select a mixture of options from a variety of available hardware and software solutions, such as “Dell PowerEdge 2900” or “Managed, Hosted Microsoft Exchange Server.” Based on a users selection, we calculate installation costs as well as monthly fees. Once the customer approves of the configuration, the system will create an invoice and notify all stakeholders, such as the administrator or account manager that an account and invoice had been created. In a separate use case, known as ‘check balance’ a user who already has an active account as well as configuration options may want to log in to review their previous billing information or check their current balance. Likewise, there is a case, known as ‘modify account’ where a user may wish to modify details of their account such as billing address, or even modify their service configuration options. Additional use cases, for example, for users to attempt password recovery, or create or delete accounts, will also be provided. Along all of the functionality contained in the Android based GUI which user interacts with, the GUI module itself will interact with a server backend, which stores our database of customer and billing information. The software will also have the ability to email or sms-text customers or administrators as required. This will work together to form a cohesive, mobile, real-time transaction processing system application centered on android that will provide an easy method for customers to select and purchase cloud based hardware and software.*

*Although our software will have the ability to receive a customer’s credit card information, no systems will be in place to process the information. Likewise, a customers credit and personal information will be stored and transmitted without encryption.*

**Use Cases:**

****

Use Case 1.1 Create Account

Primary Actor: User, Admin

Preconditions: User has an Android Device and has Installed Software.

Basic Flow of Events:

1. User requests to create account
2. Software queries user information, and creates account object
3. Software shows available Server configurations
4. User selects servers and configures options
5. Software adds server as configured to account
6. Software Notify admin

Alternative Flows:

2a. Customer does not supply information

2a1. Software does not advance to next activity

3a. Customer does not see preferred equipment in available list

3a1. Software allows for custom configurations

4. User does not select any servers

4a1. Skip step 5 and 6

Use Case 1.2 Update Account

Primary Actor: User, Admin

Preconditions: User has an account.

Basic Flow of Events:

1. User Logs into system
2. Software Presents Home Activity Screen
3. User Selects Add Server
4. User selects servers and configures options
5. Software adds server as configured to account
6. Software Notify admin

Alternative Flows:

3a. Customer selects update account information

3a1. Software shows customer information activity screen

3a2. Customer Updates information and clicks done

3a3. Software Updates Account object

3a4. Software presents home activity screen

3a5. Skip steps 4, 5 & 6

Use Case 1.3 Check Balance

Primary Actor: User

Preconditions: User has an account

Basic Flow of Events:

1. User logs into system
2. Software presents home activity screen

Alternative Flows:

1a. User cannot log into system

1a1. Software suggests Retrieve Login

Use Case 1.4 Retrieve Login

Primary Actor: User

Preconditions: User has an account.

Basic Flow of Events:

1. User requests Retrieve Login
2. Software queries DB for known account information, sets password to default
3. Software Notify user

Alternative Flows:

3a. Software does not have valid email for account

3a1. Software Notify admin

Use Case 1.5 Notify

Primary Actor: User, Admin

Preconditions: User has an account, admin has an email

Basic Flow of Events:

1. Event Trigger, such as new server created or server updated
2. Software composes text for notification
3. Software sends notification to Administrator or user as needed

**Class/Collaboration Diagram**

