# System and Software Architecture Description (SSAD)

**Hangry Mobile Food Truck Locator**

**<Team number>**

**Team Members and Roles**

John Humlick ------------------------------------------------ Architect

Amir Radman ----------------------------------------------- Project Manager/UI Designer

Aaron Escoto ------------------------------------------------ UI Designer

Daniel Tam ------------------------------------------------ Back-End Engineer

Natalia Zarubin --------------------------------------------- Back-End Engineer

**March 4**

# Version History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Author | Version | Changes made | Rationale |
| 03/04/17 | JH | 1.0 | · Create initial project draft | · Initial draft for CS 161 project |
| 5/11/17 | Neon | 1.0 | · Working prototype finished | · Presentable state of project |
|  |  |  |  |  |

# Table of Contents

System and Software Architecture Description (SSAD).................................................... i

Version History........................................................................................................................... ii

Table of Contents...................................................................................................................... iii

Table of Tables........................................................................................................................... iv

Table of Figures.......................................................................................................................... v

1. Introduction............................................................................................................................ 1

1.1 Purpose of the SSAD................................................................................................ 1

1.2 Status of the SSAD.................................................................................................... 1

2. System Analysis................................................................................................................... 2

2.1 System Analysis Overview...................................................................................... 2

2.2 System Analysis Rationale..................................................................................... 5

3. Technology-Independent Model...................................................................................... 6

3.1 Design Overview........................................................................................................ 6

3.2 Design Rationale....................................................................................................... 8

4. Technology-Specific System Design.............................................................................. 9

4.1 Design Overview........................................................................................................ 9

4.2 Design Rationale..................................................................................................... 10

5. Architectural Styles, Patterns and Frameworks......................................................... 11

# Table of Tables

Table 1: Actors Summary. 2

Table 2: Artifacts and Information Summary. 3

Table 3: Process Description. 4

Table 4: Typical Course of Action. 4

Table 5: Alternate Course of Action. 4

Table 6: Exceptional Course of Action. 4

Table 7: Hardware Component Description. 7

Table 8: Software Component Description. 7

Table 9: Supporting Software Component Description. 7

Table 10: Design Class Description. 8

Table 11: Hardware Component Description. 9

Table 12: Software Component Description. 9

Table 13: Supporting Software Component Description. 10

Table 14: Design Class Description. 10

Table 15: Architectural Styles, Patterns, and Frameworks. 11

# Table of Figures

Figure 1: System Context Diagram... 2

Figure 2: Artifacts and Information Diagram... 3

Figure 3: Process Diagram... 4

Figure 4: Conceptual Domain Model 6

Figure 5: Hardware Component Class Diagram... 6

Figure 6: Software Component Class Diagram... 6

Figure 7: Deployment Diagram... 6

Figure 8: Supporting Software Component Class Diagram... 7

Figure 9: Design Class Diagram... 8

Figure 10: Robustness Diagram... 8

Figure 11: Sequence Diagram... 8

Figure 12: Hardware Component Class Diagram... 9

Figure 13: Software Component Class Diagram... 9

Figure 14: Deployment Diagram... 9

Figure 15: Supporting Software Component Class Diagram... 9

Figure 16: Design Class Diagram... 10

Figure 17: Process Realization Diagram... 10

### 1. Introduction

#### 1.1 Purpose of the SSAD

This document is intended to define the actors, processes, and functionality of the Hangry Mobile Food Truck locator.

#### 1.2 Status of the SSAD

3/04/17: This is the first version of this document.

5/12/17: Finalized version of this document.

### 2. System Analysis

#### 2.1 System Analysis Overview

The primary purpose of Hangry Mobile Food Truck Locator is to allow users to locate local food trucks using an Android phone application. This application will track coordinates of registered food trucks and provide information -- menu, photos, and prices-- regarding specific food trucks that the user selected.

##### 2.1.1 System Context

Figure 1: System Context Diagram

Table 1: Actors Summary

|  |  |  |
| --- | --- | --- |
| **Actor** | **Description** | **Responsibilities** |
| AWS DynamoDB | A centralized DB for the application to use | Responsible for keeping and providing data stored online. |
| Google Location Services | We will be using GLS to locate food trucks in an approximate location | Responsible for providing live location of food trucks to the user. |
| Users | The users using this app to locate food truck in approximation | The responsibility of the user is to use the application for its designed purpose. |
| Food Truck Users | The food truck users | To download the app and providing permission to the app to use their live location. |

##### 2.1.2 Artifacts & Information

<< This section of the document describes the artifacts and information created by the system. These artifacts can either be used by the end user or reused by the system to produce further artifacts.

This section should contain:

- a UML class diagram showing the artifacts that the system being developed will inspect, manipulate, and/or produce

- and for each artifact shown in the UML class diagram, the artifact's name and its purpose.

More information and example can be found in **ICM EPG> Task: Analyze the Proposed System.** >>

<<Artifacts and Information Diagram>>

Figure 2: Artifacts and Information Diagram

Table 2: Artifacts and Information Summary

|  |  |
| --- | --- |
| **Artifact** | **Purpose** |
|  |  |
|  |  |
|  |  |

##### 2.1.3 Behavior

<< This section should contain:

· One or more UML Use-Case Diagrams showing the processes whereby actors and the system interact in order to accomplish a goal that benefits at least one of the actors.

· For each use case shown in the UML use case diagram(s):

* The assigned identifier (name) of the use case
* The purpose of the use case
* The requirement(s) listed in the SSRD that are (completely or partially) covered by the use case.
* The risks associated with developing/implementing the use case.
* The pre-conditions for invoking the use case
* The post-conditions that exist after the use case has been completed
* English descriptions of the use case's standard (sunny day) course of action and its alternate (rainy day) course(s) of action.

More information and example can be found in **ICM EPG> Task: Analyze the Proposed System.** >>

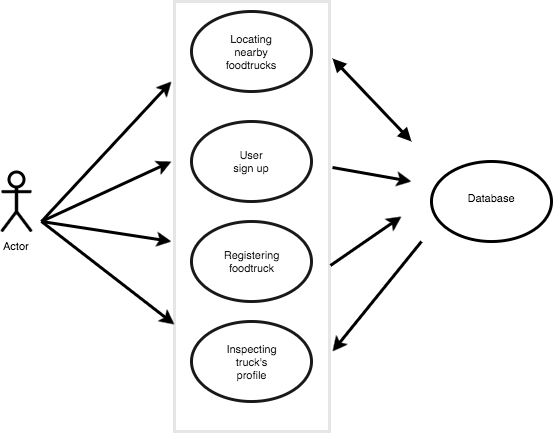


Figure 3: Process Diagram

###### 2.1.3.1 Capability: Food Truck Listings

2.1.3.1.1 Process: Listing nearby food trucks

Table 3: Process Description

|  |  |
| --- | --- |
| **Identifier** | CR-2 |
| **Purpose** | Finding nearby food trucks |
| **Requirements** | Android device with internet connectivity and a GPS |
| **Development Risks** | * Food truck drivers may not keep their locations up to date. * We may have to deal with conflicting data if we allow crowdsourced locations and gps locations |
| **Pre-conditions** | User has the Hangry Mobile Food Truck application open |
| **Post-conditions** | A google map of the local area will be displayed with an overlay of nearby food truck |

Table 4: Typical Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User opens the app. | App obtains the user’s gps coordinates. |
| **2** | User specifies the search radius for nearby food trucks (e.g. 5 miles). | App queries Amazon AWS DynamoDB for all listings within the specified range. |
| **3** | App sends all locations returned from AWS/dynamodb to Google Location Services. | A list view or a map view (depending on orientation) is displayed showing all food trucks within the specified search radius. |

Table 5: Alternate Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User opens the app. | App obtains the user’s gps coordinates. |
| **2** | User searches for the name of a food truck. | App queries Amazon AWS dynamodb for all listings containing the specified food truck name. |
| **3** | App sends all locations returned from AWS/dynamodb to Google Location Services. | A list view or a map view (depending on orientation) is displayed showing all food trucks matching the specified name. |

Table 6: Exceptional Course of Action

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | User opens the app. | App tries to obtain the user’s gps coordinates, but cannot. |
| **2** | App displays an option to enable location services if it is disabled, grant permission to the GPS permissions are not granted, or an error is displayed if the location cannot be obtained. | App waits for corrective action from the user, then tries again. |

##### 2.1.4 Modes of Operation

<< For projects involving systems that operate in more than one mode this section should contain a description of the system's behavior in each of its modes. More information and example can be found in **ICM EPG> Task: Analyze the Proposed System.** >>

The primary goal of this application is to provide the user with the current location of the food truck, if the user or truck driver lose connection, the application will temporarily be disabled and will resume once the connection is re-established. Therefore, there will be only one mode of operation.

#### 2.2 System Analysis Rationale

<< This section should list and explain aspects of the analysis that are deemed by the team to be less than obvious or actually counter-intuitive and for which, as a result, there is a high risk that readers will not understand or will misunderstand what is intended. More information and example can be found in **ICM EPG> Task: Analyze the Proposed System.** >>

Cases where users can misunderstand how to use the application:

* The truck drivers may not know they need to broadcast their location or have the icon running in the background.
* User accidentally selects the portion to rate a food truck unintentionally.

### 3. Technology-Independent Model

#### 3.1 Design Overview

##### 3.1.1 System Structure

<< This section should contain

· a conceptual domain model

· a UML hardware component class diagram

· a UML software component class diagram

· a UML deployment diagram

· If necessary, a class diagram for the system's supporting software infrastructure

· and descriptions of the hardware components, software components, and, if necessary, the supporting software infrastructure components of the technology/platform-independent system architecture

More information and example can be found in **ICM EPG> Task:** **Define Technology-Independent Architecture >>**

<<Conceptual Domain Model>>

Figure 4: Conceptual Domain Model

<<Hardware Component Class Diagram>>

Figure 5: Hardware Component Class Diagram

<<Software Component Class Diagram>>

Figure 6: Software Component Class Diagram

<<Deployment Diagram>>

Figure 7: Deployment Diagram

<<Optional: Supporting Software Infrastructure Diagram>>

Figure 8: Supporting Software Component Class Diagram

Table 7: Hardware Component Description

|  |  |
| --- | --- |
| **Hardware Component** | **Description** |
|  |  |
|  |  |
|  |  |

Table 8: Software Component Description

|  |  |
| --- | --- |
| **Software Component** | **Description** |
|  |  |
|  |  |
|  |  |

Table 9: Supporting Software Component Description

|  |  |
| --- | --- |
| **Support Software Component** | **Description** |
|  |  |
|  |  |
|  |  |

##### 3.1.2 Design Classes

This section should contain:

· UML class diagrams showing all the boundary, entity, and control classes in the design of the system being developed

· and a description of each class in the diagram

More information and example can be found in **ICM EPG> Task:** **Define Technology-Independent Architecture >>**

###### 3.1.2.1 <Classes n>

<<Design Classes Class Diagram>>

Figure 9: Design Class Diagram

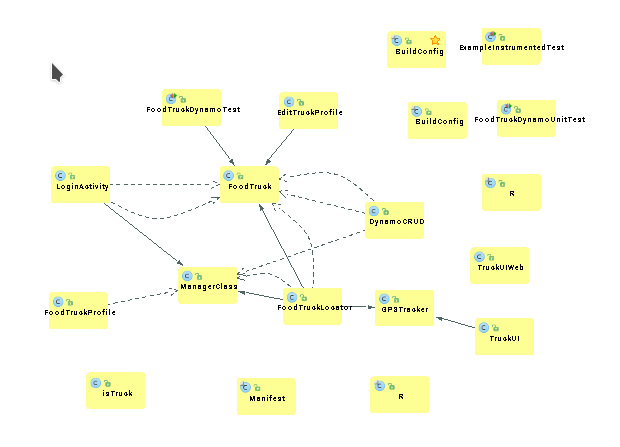


Table 10: Design Class Description

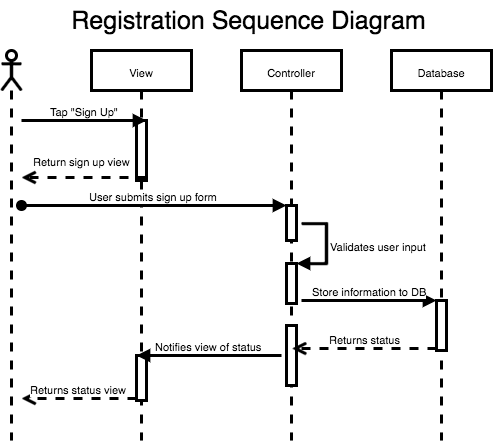
|  |  |  |
| --- | --- | --- |
| **Class** | **Type** | **Description** |
|  |  |  |
|  |  |  |
|  |  |  |

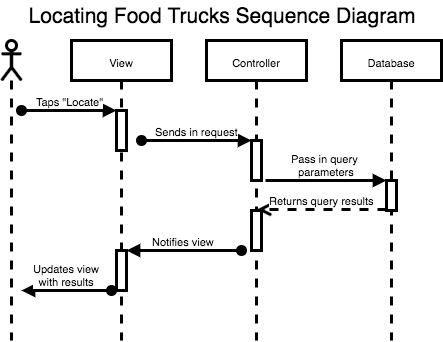
##### 3.1.3 Process Realization

<< This section shows how the proposed architecture can be realized by conducting robustness analysis and constructing sequence diagrams. More information and example can be found in **ICM EPG> Task:** **Define Technology-Independent Architecture >>**

<<Robustness Diagram>>

Figure 10: Robustness Diagram





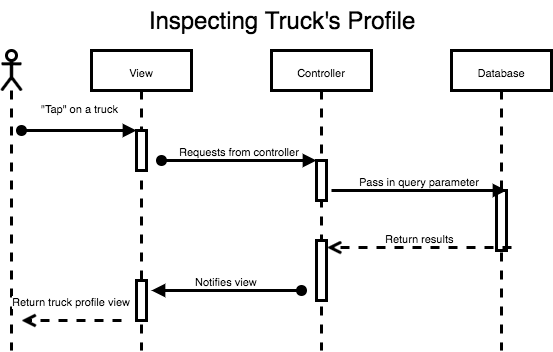


Figure 11: Sequence Diagram

#### 3.2 Design Rationale

<< This section should contain an explanation of how/why the architecture/design described in previous sections was chosen. More information and example can be found in **ICM EPG> Task:** **Define Technology-Independent Architecture >>**

### 4. Technology-Specific System Design

<< Once you know specific technology that you team is going to use, design the system and software architecture and document them in this section. >>

#### 4.1 Design Overview

##### 4.1.1 System Structure

<<Hardware Component Class Diagram>>

Figure 12: Hardware Component Class Diagram

<<Software Component Class Diagram>>

Figure 13: Software Component Class Diagram

<<Deployment Diagram>>

Figure 14: Deployment Diagram

<<Optional: Supporting Software Infrastructure Diagram>>

Figure 15: Supporting Software Component Class Diagram

Table 11: Hardware Component Description

|  |  |
| --- | --- |
| **Hardware Component** | **Description** |
| Android Phone | Phone must have GPS and Wifi. |
|  |  |
|  |  |

Table 12: Software Component Description

|  |  |
| --- | --- |
| **Software Component** | **Description** |
| Android | Phone must be on Android 4.0 or higher. |
|  |  |
|  |  |

Table 13: Supporting Software Component Description

|  |  |
| --- | --- |
| **Support Software Component** | **Description** |
| AWS Dynamo/Lambda | AWS to handle our cloud storage and triggers. Stores user profiles |
| Google Play Services | Google services allows us to get food truck coordinates |
|  |  |

##### 4.1.2 Design Classes

###### 4.1.2.1 <Classes n>

<<Design Classes Class Diagram>>

Figure 16: Design Class Diagram

Table 14: Design Class Description

|  |  |  |
| --- | --- | --- |
| **Class** | **Type** | **Description** |
|  |  |  |
|  |  |  |
|  |  |  |

##### 4.1.3 Process Realization

<<Process Realization Diagram>>

Figure 17: Process Realization Diagram

#### 4.2 Design Rationale

### 5. Architectural Styles, Patterns and Frameworks

<< Describe any implementation architecture styles (e.g. the Prism style and 3-tier architecture), patterns (e.g. pipe-and-filter and client-server), or frameworks (e.g. Java and CORBA) used to describe the system architecture. >>

Table 15: Architectural Styles, Patterns, and Frameworks

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Benefits, Costs, and Limitations** |
| Model View Controller | Separates how the data gets handled between the layers | Improves organization, limit user access to database, restricts unauthorized activities |
|  |  |  |
|  |  |  |