**Software Architecture Document**

**Table of Contents**

**1. Introduction**

**1.1 Purpose**

**1.2 Scope**

**1.3 Definitions, Acronyms and Abbreviations**

**1.4 References**

**2. Architectural Representation**

**3. Architectural Goals and Constraints**

**4. Use-Case View**

**4.1 Architecturally-Significant Use Cases**

**5. Logical View**

**5.1 Class Diagram**

**6. Process View**

**6.1 Processes**

**6.2 Activity Diagram**

**6.3 Concurrency and Synchronization**

**7. Development View**

**7.1 Component diagram**

**7.2 Package Diagram**

**8. Deployment View**

**1. Introduction**

#### **1.1 Purpose**

1. **The purpose of the document is to provide a comprehensive architectural overview of the system that is ‘*Parking Lot Assistant’* using a number of different architectural views in order to depict the different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.**

#### **1.2 Scope**

1. **This Software Architecture Document provides an architectural overview of the ‘*Parking Lot Assistant’*. The Parking Lot Assistant System is being developed to assist the drivers in order to get a free parking lot within a parking zone without wasting time to search for the space.**

#### **1.3 Definitions, Acronyms and Abbreviations**

1. **See the Glossary [4].**

#### **1.4 References**

**2. Architectural Representation**

1. **This document presents the architecture as a series of views; use case view, logical view, process view and deployment view. There is no separate implementation view described in this document. These are views on an underlying Unified Modeling Language (UML) model developed using Rational Rose.**

**3. Architectural Goals and Constraints**

**There are some key requirements and system constraints that have a significant bearing on the architecture. They are:**

* 1. **Architecture Goal 1**
  2. **Architecture Goal 2**

**4. Use-Case View**

**A description of the use-case view of the software architecture. The Use Case View is important input to the selection of the set of scenarios and/or use cases that are the focus of an iteration. It describes the set of scenarios and/or use cases that represent some significant, central functionality. It also describes the set of scenarios and/or use cases that have a substantial architectural coverage or that stress or illustrate a specific, delicate point of the architecture.**

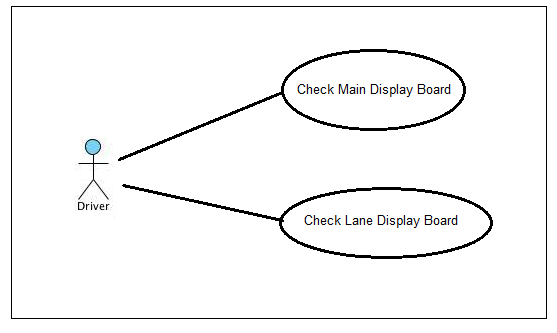
**The Use cases are**

1. **Use Case1**
2. **Use Case2**

**4.1 Architecturally-Significant Use Cases**

**Diagram Name: Architecturally Significant Use-Cases**

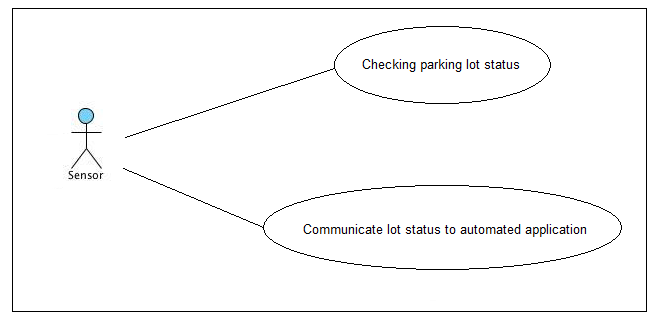
***4.1.1 Use Case 1***



**Brief Description:**

1. **This use case allows the driver of a car to check the main display board to check the availability of free space in a parking lot. If the total space is more than the number of cars in the queue, then the car enters the parking zone.**
2. **The driver after entering the parking zone checks each of the lane display board to check the number of free parking lots.**

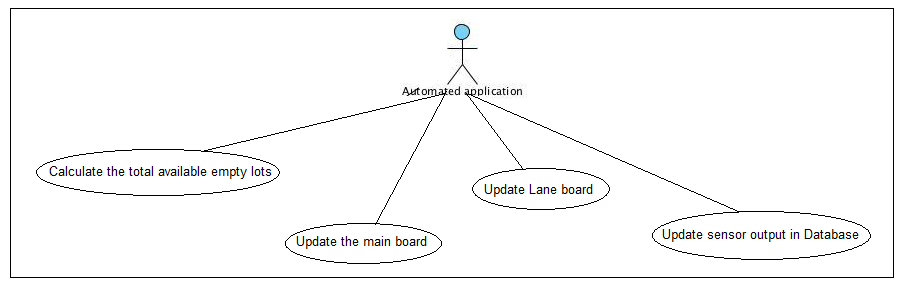
***4.1.2 Use Case 2***



**Brief Description:**

1. **The sensor checks if a particular parking lot is occupied by a car.**
2. **The sensor sends information to the database if a particular parking lot is occupied by a car.**

***4.1.3 Use Case 3***



**Brief Description: This use case helps a central application system to control and manage the parking space using the data from the sensors.**

1. **It gets regular updates from the sensor, whether a particular parking lot is occupied by a car.**
2. **On the basis of the data received from the sensor, the central application counts the number of available space in a parking zone.**
3. **According to the calculation done, it updates the main board.**
4. **The application also sends the exact location that is the the lane which has got free parking zone.**

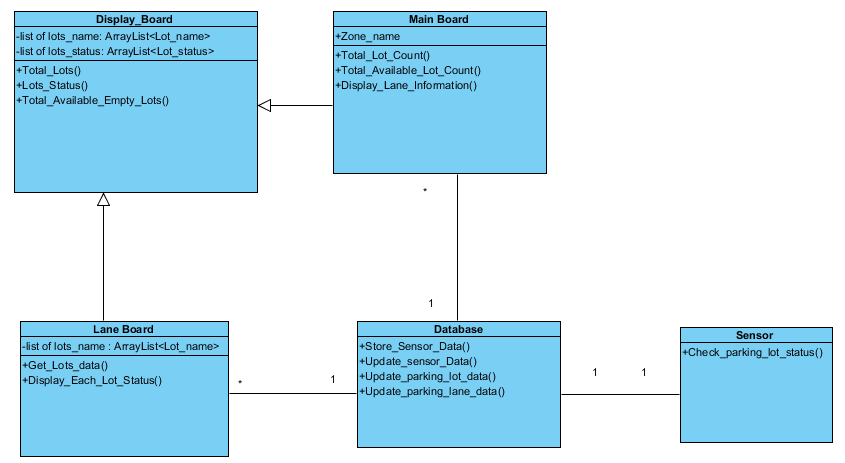
**5. Logical View**

**The services provided by the system to the end user are the following :**

* **End user will be able to know the list of parking lots having available spaces**
* **End user will be able to know the availability of space in each lot**
* **End user will be able to know the parking lane information**

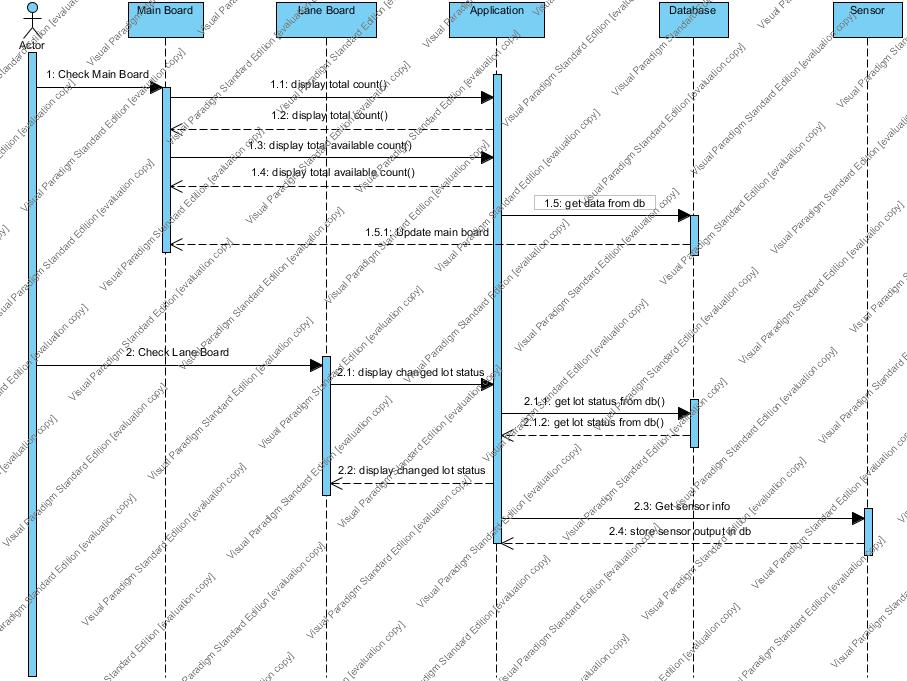
**5.1 Class Diagram**

**In a Class Diagram, The subparts of display board - main board and lane board are the user screens. Main Board and lane board retrieve the data from database. Database stores sensor data and updates the available information. Sensor checks the status for available space in parking lot area.**



**6. Process View**

**6.1 Processes**



**Diagram Name: Process Sequence Diagram**

***6.1.1* Main Board**

**The main board will display a given zone’s total available lots. Ideally this would be placed at the beginning of the zone.**

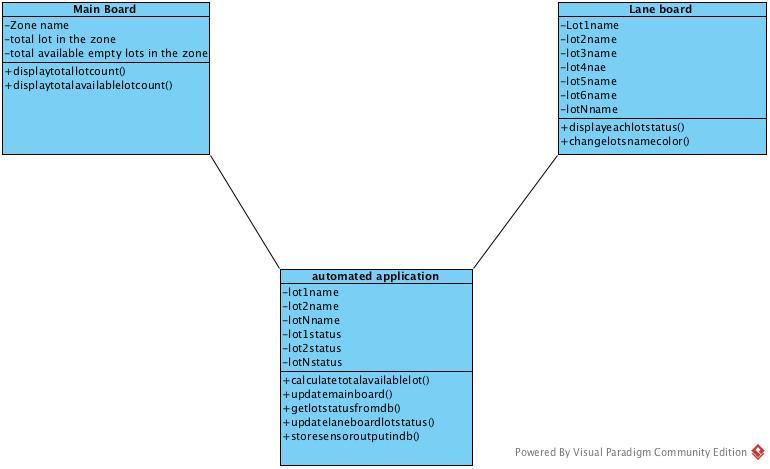
***6.1.2*  Automated Application**

**Automated application is the entity that monitors and manages the activity throughout the parking lots, parking lanes and individual parking spots. The application keeps track of the available and occupied spaces in a lane and displays the related information on a lane board.The application gathers all the data from the lanes in the parking lot and display the on the main board.**

***6.1.3*  Lane Board**

**The lane board will display a given lane’s (inside a zone) total available lots. Ideally this would be placed at the front of every lane.**

**6.2 Activity Diagram**



**Diagram Name: Processes**

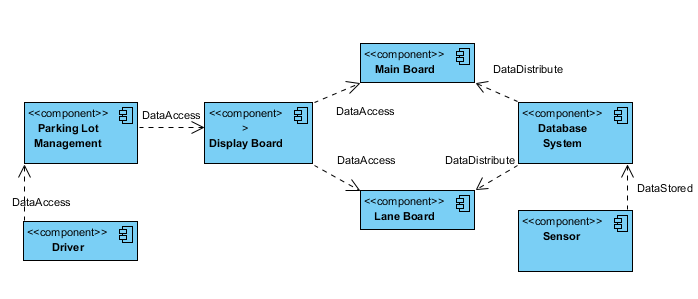
**6.3 Concurrency and Synchronization**

**There are no major concurrency issues to be considered. The only issue will be that the display boards will need to be continually synchronized based on the ever changing contents of the database. This will be achieved by querying the database in a regular time interval. All processes can be considered modular and do not require interprocess communication.**

**7. Development (Implementation) View**

**The application is decomposed to several subsystems (components), including the user interface system which can be further decomposed to main display board and the parking lot lane display board, the database system and sensor system. Each component can be assigned to an individual developer and be implemented independently. Two UML diagrams have been developed to represent the development view.**

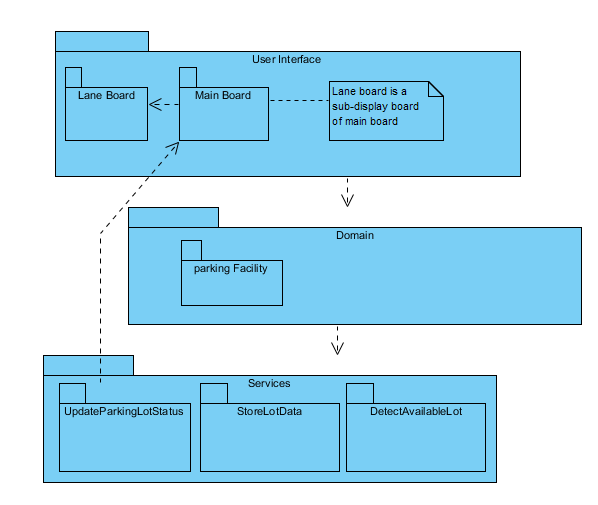
**7.1 Component diagram**



**Brief description:**

1. **Driver can have knowledge of the parking lot availability status through the application interface.**
2. **The application access the data through display board which contains the Main board and Lane board.**
3. **Sensor is responsible for checking lot status and stored the status information to the database for later use.**
4. **Database is responsible for distributing saved lot data to the display board.**

**7.2 Package diagram**



**8. Deployment View**

1. **In order to do implement and deploy this software, we have to make a few abstractions. First the sensors described in the specification document, will be emulated via a program that will randomize each lots occupancy. Second, each board described in will be displayed as a web-based UI. As there is no physical display board that could be used that would fit the scope of this project. The results from the emulated sensors will be stored in the database. The database will be queried and display aggregated lot data on the UI. We will assume that the lot data could be displayed on a display or monitor in appropriate location in the parking zone and lane.**
2. **The database will be set up on Amazon RDS (using a MySQL database). The web-based application will also be available using Amazon EC2. This instance will be configured with Apache to appropriately route requests for the application and setup a connection with the database. The deployment and configuration will be automatic by using Chef and writing a recipe for the application to install all necessary parts. Each Amazon instance will run on the smallest instance provided by Amazon (micro).**
3. **Diagram Name: Deployment View**

