Self Service Car Wash System

## Software Architecture and Technical Design

### Dinu Ionela Veronica

# Purpose

The main goal of this project is to enable the functionality of a self-service car wash business, with the lowest investment possible.

After a quick study of the market, we’ve discovered that this is a highly demanded service, mostly in highly populated cities, due to the increasing number of vehicles.

The main goal would be to enable the end users to be able to self-service: automated machine near the pumps that allows them to pay, activate and use the washing system.

# Business Requirements

The following list is describing the business requirements for the system.

* As a Business Owner, I want to have 3 systems that allows clients to pour water and other liquids onto their cars, using high pressure. The systems would have to be automatically controlled by a supervising entity.
* As a Business Owner, I want to have 3 systems that allows clients to be able to self-service before and during the usage of the Wash System. This must include payment, selection of the type of service and control. Each system should be tightly coupled with a corresponding liquid delivery system

# Feasibility Analysis

By analyzing the business requirements, we didn’t identify any potential hardware or software impediments.

# Proposed Solution

As a solution, we propose a system built with the following specifications:

* Liquid Delivery System
  + 6 X high pressure liquid pumps = $6000
  + 6 X GPIO relays = $6
  + 15 X liquid containers = $1500
* Self Service Box
  + 6 inch LCD screen with RCA connectivity = $50
  + Coin (token) collection system with GPIO connectivity = $100
  + Slimmed down USB keyboard with only: 1,2,3,4,5,6,7,8,9,STOP keys = $5
  + Raspberry PI model 4 = $35
  + .NET 6 Controller software implementation – 80 hours x 1 developer = $1600
* Architecture
  + 10 hours x 1 architect = $5000

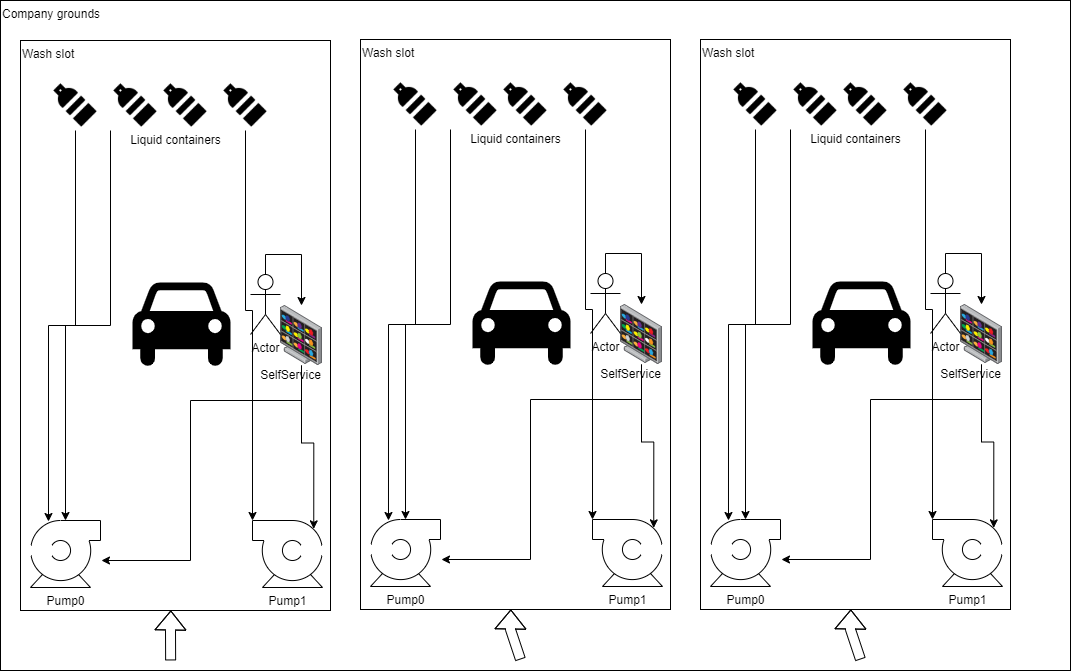
The total amount spent is very low compared to projected earnings.

# Agile Template

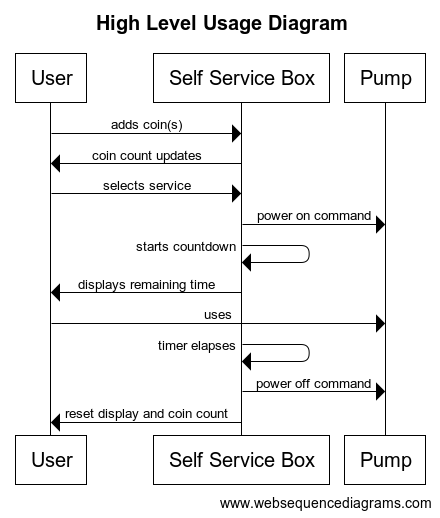
By analyzing the Business requirements, we can extract a set of Themes, Epics and User Stories, as follows:

* Architecture and design
  + Hardware
    - Liquid Delivery System
    - Self Service Box
  + **Software (covered here)**
    - High Level Architecture
    - Detailed Architecture (UML)
    - High Level Flow Sequence Diagrams
    - Detailed Sequence Diagrams
* Liquid Delivery System
  + Pumps
    - Purchase
    - Installation
  + Relays
    - Purchase
    - Installation
    - Integration with Pumps
  + Liquid Containers
    - Purchase
    - Installation
    - Integration with Pumps
* Self Service Box
  + Hardware build for Raspberry PI
    - LCD installation
    - Coin collection system installation
    - Keypad installation
    - Pump relay installation
  + Software build
    - Raspberry PI setup
    - Integration with LCD
    - Integration with Keypad
    - Integration with pump relays
    - Integration with coin collection system
    - Unit testing
    - Builds & deployment

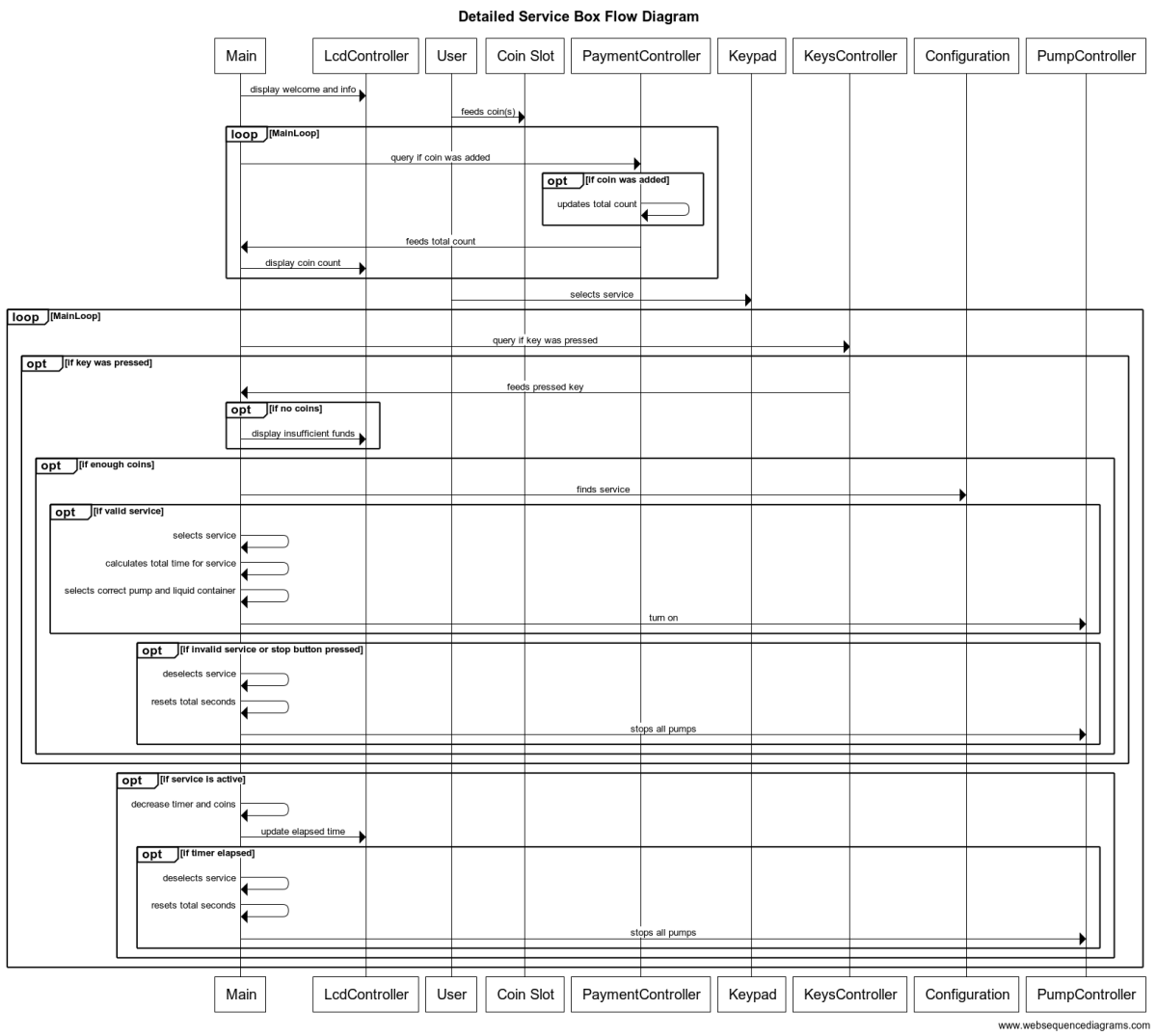
# High Level Architecture Diagram



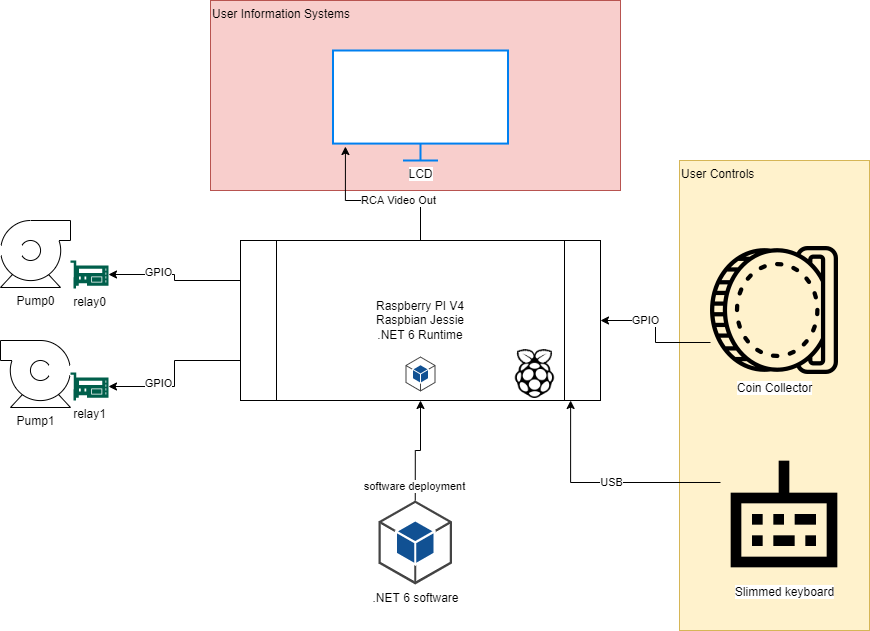
# High Level Service Box Usage Sequence Diagram



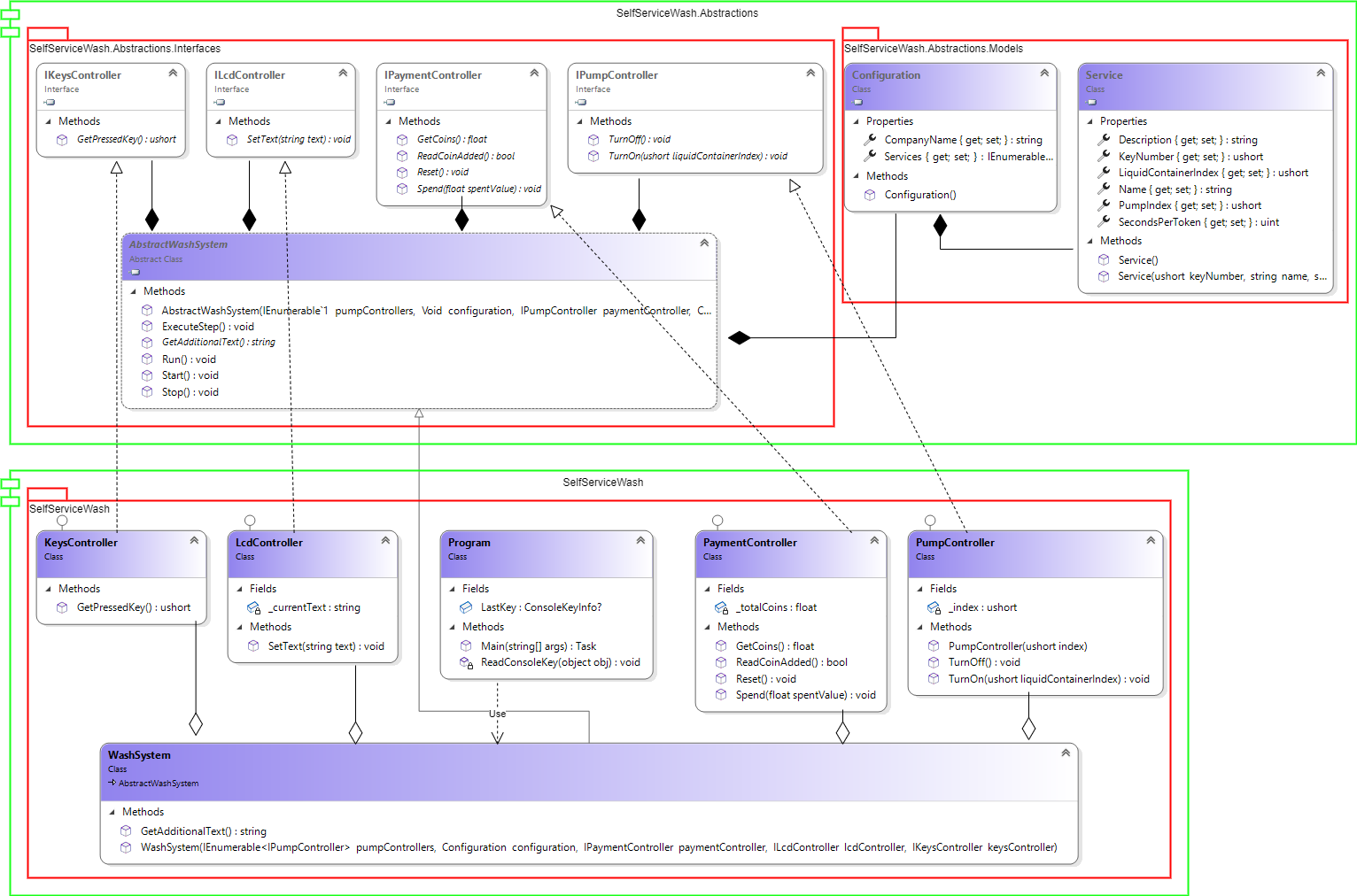
# Detailed Service Box Flow Sequence Diagram



# Self Service Box Software Architecture Diagram



# Self Service Box Software UML Class Diagram



# Self Service Box Software implementation details

* Programming language: .NET 6 (C# 10) - it is cross-platform, free, and it supports ARM x64 (Raspberry PI CPU)
* System.GPIO library to be used
* Visual Studio 2022 Community as IDE (free)
* XUnit for unit tests