Rentd

Car rental platform

Badhax

BADHAX INC

1. Introduction

1.1. Purpose of Document

The purpose of this document is to technically outline the features and architecture of the proposed system.

1.2. Scope

The scope of this work is the implementation of Rentd which will be a web application. This application will bridge the gap between motor vehicle owners who has unutilized vehicles and persons wishing to acquire temporary usage of said vehicles.

2. System Objectives

This solution is focused on enabling users to list their vehicle in a high traffic platform that will increase the visibility of their offering and potential buyers can find the vehicles that best matches their needs.

The functional areas of the system are:

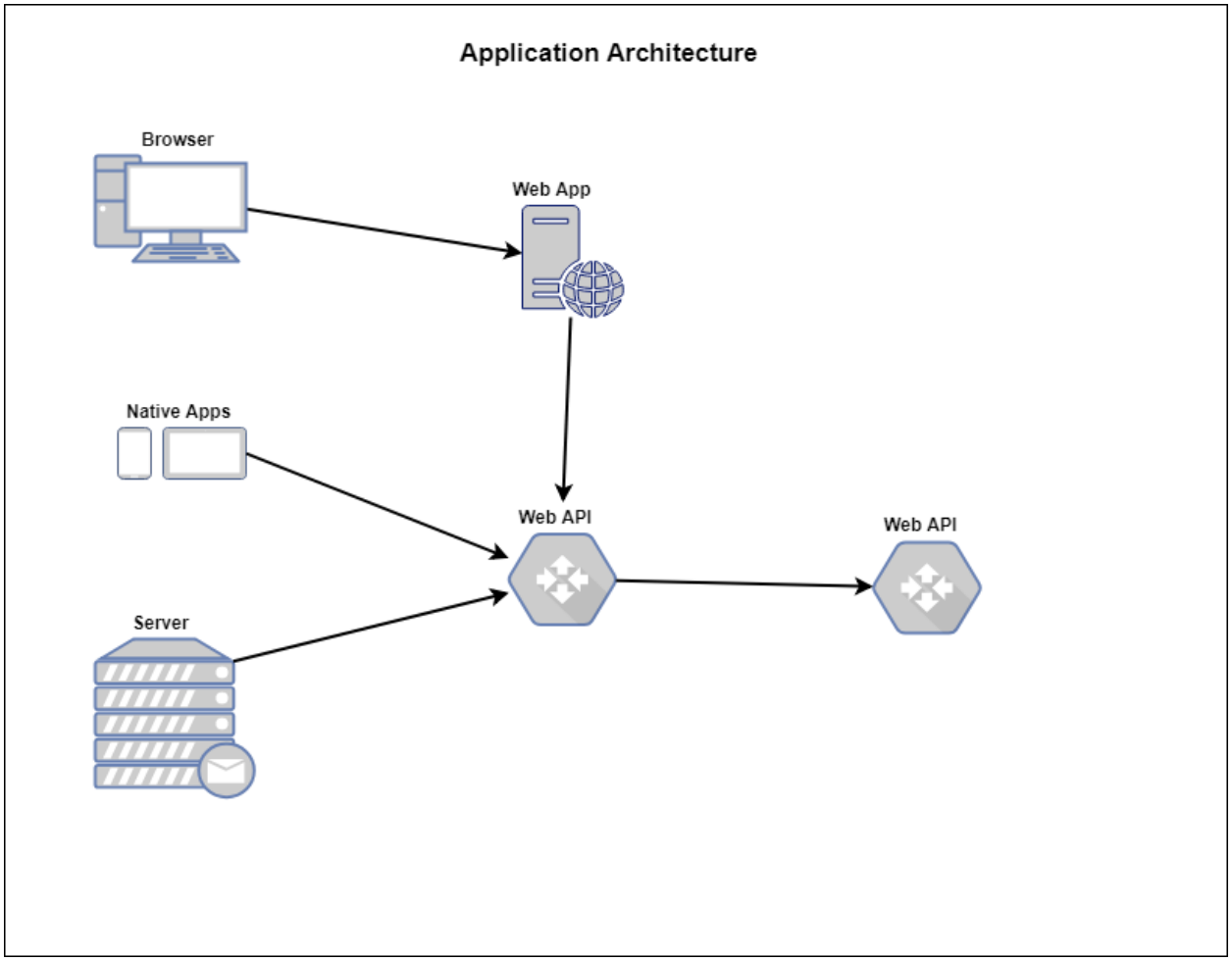
- External Login
    - Users should be able to login using different third-party accounts
        - Facebook
        - Twitter
        - Google
        - Microsoft
- Listing of vehicles
    - Renters will be able to list details of the vehicle
        - Must include a valid picture
        - User must enter vehicle registration details
- Verification of ownership and insurance
- Verification of driver's license eligibility
- Category Selection
    - Users can select from three classes of vehicles:
        - Cars
        - Mini vans
        - Trucks
- Match Filtering
    - Users Can filter the closest offers
    - Users Can filter by the most accurate offer
- Payment
- Additional Services

3. Technical Overview

The application will follow a modern client-server design, where:

- Browsers will communicate with the web application
- The web application will complete functions using web APIs (On its own and as user request data)
- Native Applications will leverage the same web APIs as the web app
- The application web APIs will communicate with external Web APIs (to leverage published functionality; both paid and free)

The communication should look like this:



Architecture

The client:

- Angular 5
- Angular CLI
- Angular Material

Server:

- IdentityServer4 (OIDC)
- .Net Core 2.1 API
- Entity Framework code first Migration
- SQL Server DB

Security

The application will be using a claim-based framework for authentication of the different parts.

This will allow us to have:

- Single sign on
    - Users can sign out of the by signing out of the web client the user will also be signed out of any other device and vice-versa.
- Use an open source identity provider
    - Rather than implement a standalone identity provider which will have to be maintained personally, we use a supported identity framework.
- Mobile compatibility
- Security for our APIs