Софийски университет „Св. Климент Охридски“
Факултет по математика и информатика

Курсова работа

по

Софтуерни архитектури и разработка на софтуер

на тема

VeloCity – система за отдаване на велосипеди под наем

Изготвена от:

Александър Спасов, 62386

Васил Христов, 62431

Теодор Дяков, 61967

Съдържание

1. Въведение

1.1. Обща информация за текущия документ

1.1.1. Предназначение на документа

Предназначението на документа е да се представи софтуерната архитектура на система за отдаване под наем на велосипеди в рамките на града(VeloCity).

1.1.2. Описание на използваните структури на архитектурата

1.1.2.1. Декомпозиция на модулите

Тази структура показва системата, разделена на отделни модули. Модулите са: Client Mobile App, Monitoring Web App, VeloCity Back-end, Database, Smart Bike, Payment API, Map API. Client Mobile App е приложението, което свързва потребителя със системата. Уеб приложението е предназначено за използване от системните администратори и наблюдателите. Back-end частта на системата взаимодейства с другите модули, за да може системата да работи според изискванията. Базата данни съхранява всички данни в системата и предоставя достъп до тях. Smart Bike се състои от софтуера и хардуера, който ще бъде внедрен върху самото колело. Payment API е външен приложен програмен интерфейс, предоставен от система за извършване на онлайн плащания. Map API е външен приложен програмен интерфейс, който се предоставя от доставчик на услуги за карти и геолокация.

1.1.2.2. Структура употреба на модулите

Различните статични връзки между отделните модули и подмодули на системата се визуализират от тази структура.

1.1.2.3. Структура на процесите

Изобразява нагледно някои от важните процеси на системата.

1.1.2.4. Структура на внедряването

Структурата на внедряването ни показва разположението на софтуера върху хардуера.

1.1.3. Структура на документа

Документът се състои от 4 секции:

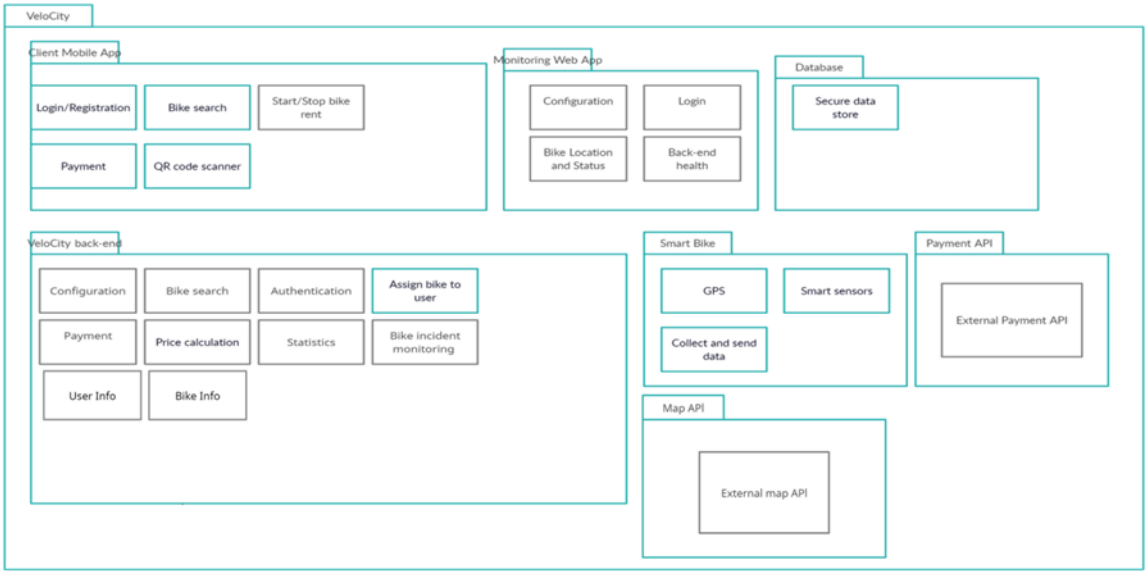
1) Въведение към документа;
2) Структура Декомпозиция на модулите;
3) Допълнителни структури(Употреба на модулите, структура на процесите, структура на внедряването);
4) Обосновка на архитектурата, където се определят архитектурните драйвери и се обясняват причините те да бъдат избрани.

1.2. Общи сведения за системата

VeloCity е система за отдаване под наем на велосипеди в рамките на града. Основната цел на системата е потребителите през мобилно приложение да се информират за най-близките свободни велосипеди, с които може да се придвижват само в рамките на града. Велосипедите са снабдени с електрическа система за задвижване, но може да се задвижват и със собствена сила на колоездача, чрез педали.

2. Декомпозиция на модулите

2.1. Общ вид на Декомпозиция на модулите



2.2. Списък на модулите

1. VeloCity
1.1. Client Mobile App
1.1.1. Login/Registration
1.1.2. Bike search
1.1.3. Start/Stop bike rent
1.1.4. Payment
1.1.5. QR code scanner
1.2. Monitoring Web App
1.2.1. Configuration
1.2.2. Login
1.2.3. Bike location and status
1.2.4. Back-end health
1.3. Database
1.3.1. Secure data store
1.4. VeloCity Back-end
1.4.1. Configuration
1.4.2. Bike search
1.4.3. Authentication
1.4.4. Assign bike to user
1.4.5. Payment
1.4.6. Price calculation
1.4.7. Statistics
1.4.8. Bike incident monitoring
1.4.9 User Info
1.4.10 Bike Info
1.5. Smart Bike
1.5.1. GPS
1.5.2. Smart sensors
1.5.3. Collect and send data
1.6. Payment API
1.6.1. External Payment API
1.7. Map API

2.3. Описание на модулите

2.3.1. VeloCity е основната система.

2.3.1.1.Мобилно приложение за клиенти

Този модул, който се грижи за мобилното приложение за клиенти се състои от подмодули, които определят функционалностите на системата от гледната точка на клиента.

2.3.1.1.1. Вход в системата/Регистрация

Регистрацията в системата на VeloCity е безплатна за всяко лице. Задължително трябва да се въведат следните лични данни: име, презиме, фамилия, ЕГН, данни за връзка като телефонен номер и e-mail адрес, както и потребителско име и парола за влизане в системата.
След успешна регистрация потребителите ще могат да влизат в системата след като въведат потребителското име и паролата си.

2.3.1.1.2. Търсене на велосипед

Отговаря за лесното и бързо намиране на велосипед при заявка на потребителите на системата, като се търсят тези велосипеди с поне X% батерия и които се намират на най-близката до местоположението на потребителя стоянка.

2.3.1.1.3. Стартиране/Спиране на наемането на велосипед

При желание на потребителя да вземе някой велосипед, той трябва да отбележи в приложението велосипеда, който заема, както и начина за плащане на услугата. След това потребителят може да използва велосипеда докато максималното време T не изтече. Когато времето изтече, наемателят получава съобщение, което го информира, че времето му е изтекло и трябва да остави велосипеда на най-близката до него стоянка.

2.3.1.1.4. Заплащане

Регистриран потребител, който е готов да наеме някой велосипед, може да заплати услугата по три различни начина: чрез СМС, с кредитна карта или като предварително си закупи талон с QR код.

2.3.1.1.5. Сканиране на QR кодове

При избор на потребителя да заплати услугата на VeloCity, като си закупи талон с QR код, системата позволява два начина за въвеждане на кода:

1) Ръчно – кодът се пише от потребителя;
2) Автоматично – кодът се сканира от системата с помощта на камерата на мобилното устройство.

2.3.1.2.Уеб приложение

Уеб приложението е предназначено за използване от системните администратори и наблюдателите(отговорниците по използването на велосипедите).

2.3.1.2.1. Конфигурация

Този модул определя правомощието на системните администратори да конфигурират системата и да следят за правилното ѝ функциониране.

2.3.1.2.2. Вход в системата

След като са се регистрирали успешно администраторите и наблюдателите, те могат да влязат в системата като въведат избраните от тях потребителско име и парола.

2.3.1.2.3. Местоположение и статус на велосипеда

Всеки велосипед има собствен идентификационен номер в системата и е снабден с GPS и смарт-сензори за диагностика, които позволяват на системните администратори постоянно да следят къде се намира всеки велосипед, както и какво е неговото състояние(дали е свободен, дали е нает или дали е попаднал в някакъв инцидент).

2.3.1.2.4. Back-end health

Тук се има за задача да се дава информация за състоянието на системата по всяко време от денонощието, за да може администраторите да бъдат информирани за евентуални грешки при натоварвания, както и при заплахи за системата и да могат бързо да реагират и да ги оправят.

2.3.1.3.База данни

Този модул е за съхранение на данните за велосипедите, потребителските данни (в защитена от външна намеса среда),както и за останалите данни на системата.

2.3.1.3.1. Secure data Store

Тук в безопасна среда се съхраняват личните данни на потребителя. Достъп до данните имат единствено хората с роля на наблюдател на правомерното използване на велосипедите.

2.3.1.4.VeloCity Back-end

Този модул е за back-end частта на системата. Тази част на системата е недостъпна за потребителя. Тя взаимодейства с другите модули, за да може системата да работи според изискванията.

2.3.1.4.1. Configuration

Този модул дава достъп на системните администратори, а също и на наблюдателите на системата и определя техните правя. Наблюдателите на системата получават достъп и до данните на потребителите, за да могат да осигурят безопасно използване на системата.

2.3.1.4.2. Bike search

Модулът има задачата да предостави местоположението на най-близките велосипедите, отговарящи на изискването да имат >=X% батерия. Трябва да се използва добър алгоритъм за намиране на велосипеди, удовлетворяващи изискванията в радиуса на потребителя.

2.3.1.4.3. Authentication

Тук се проверяват данните на потребителя и се прави проверка за валиден имейл при регистрация на нов профил във VeloCity. При заплащане със кредитна карта се проверява датата на изтичане на картата, номер , и при нужда се изпраща онлайн 3D парола за потвърждение на плащането по СМС. В случай на използване на QR-code се проверява валидността му и автентичността му.

2.3.1.4.4. Assign bike to user

Модулът предоставя достъп до велосипеда на VeloCity при намиране на велосипед в близост до регистриран потребител и при проверка на извършено плащане. При премината валидация на плащането започва да тече таймер с максимално време на използване на велосипеда – T.

2.3.1.4.5. Payment

При избор на подходящ велосипед от потребителя му се предоставя опцията за плащане. Възможностите за плащане са чрез СМС, кредитна карта или чрез уникален QR-code. При избор на една от тези три възможности, и при

въвеждане на данните за плащане се преминава към валидация на плащането. Когато валидацията е успешна се изпраща известие на потребителя, че вече са отключени функциите на велосипеда и таймерът за време започва да тече.

2.3.1.4.6. Price calculation

Отговорността на модула е да изчисли сумата, която трябва да бъде платена, като се съобрази времето за ползване на колелото.

2.3.1.4.7. Statistics

В този модул се създават статистики за ползването на велосипедите, за процента различни плащания, за инцидентите възникнали в определен ден, както и други статистики, които биха помогнали за подобряване на системата.

2.3.1.4.8. Bike incident monitoring

Тук се следи за възникнали инциденти възникнали по време на ползване на велосипед на VeloCity. При възникване на пътен инцидента автоматично се изпраща сигнал до 112,в рамките на 1 секунда след засичане на инцидента. До 5 секунди се известява наблюдателя на системата.

2.3.1.4.9. User Info

Предназначението на този модул е дава е да служи като интерфейс, който дава информация за потребителите в приложението. Чрез него може да бъде извличана информация за парола, име, ЕГН на даден потребител и други.

2.3.1.4.10. Bike Info

Предназначението на този модул е да предоставя информация за колелата в системата. Информацията за колело може да бъде търсена по уникалния идентификационен номер на велосипеда. Този модул предсотавя цялостна информация за един велосипед включително заетост(дали е нает в момента), GPS координати както и данните от сензорите за последните няколко часа от велосипеда.

2.3.1.5.Smart Bike

Този модул се състои от софтуера и хардуера, който ще бъде внедрен върху самото колело.

2.3.1.5.1. Smart sensors

Този модул показва внедрените умни сензори върху колелото. Тяхната функция е да събират информация за различни показатели, която поне да може да бъде използвана. Върху колелото трябва да се внедрят поне 3 различни сензора: Трябва да присъства сензор, който да събира информация за налягането в гумите на велосипеда. Информацията от този сензор, ще може да бъде използвана от екипите по техническа поддръжка, за да могат да разберат кои велосипеди имат нужда от техническа интервенция. Освен това трябва да има сензор, който да може да улавя резки промени в ускорението. Информацията получена от този сензор ще послужи за да може да бъдат разпознавани инциденти, случвали се по време на управлението на велосипеда. Това може да бъде реализирано с така наречение shock sensor. Нужно е да има сензор който да може да дава информация състоянието на батерията. При нея може да се използва сензор, измерващ физически характеристики на батерията. Най-важните физични величини, които трябва да може да бъдат измервани са големината на тока, напрежението и температурата на батерията.

2.3.1.5.2. GPS

Върху колелото е необходимо да има GPS модул, който да предоставя информация за географската позиция на колелото. Тази информация ще бъде използвана от системата, за да може да следи географското местоположение на всички велосипеди.

2.3.1.5.3. Collect and send Data

Този модул ще служи за да може да събира данните от умните сензори и да ги изпраща към тези модули на VeloCity които ще запазват и обработват данните от сензорите. Модулът ще чете данни от сензорите, ще ги пакетира във удобен за пренос през мрежата вид и ще ги изпраща към обработващия модул. Един начин за реализация на този модул е със използването на вградена система, например Raspberry, която е свързана към

интернет. Друг вариант е вградения компютър да изпраща данните към телефона на потребителя чрез Bluetooth или Wifi-direct. По този начин предимството е че няма да има нужда вградения компютър да бъде свързан към интернет.

2.3.1.6.Payment API

Външен приложен програмен интерфейс, предоставен от система за извършване на онлайн плащания. Ще се използва за осъществяване на плащане на дължимата такса след използване на велосипед. Плащането ще се извършва посредством онлайн транзакция след като клиента предостави необходимата информация за плащането. Възможно е да има повече от едно API, тъй като в изискванията системата са споменати няколко различни начина за плащане.

2.3.1.7.Map API

Външен приложен програмен интерфейс, който се предоставя от доставчик на услуги за карти и геолокация. Чрез информацията предоставена от този интерфейс ще може да се търсят най-близките Тази информация ще бъде използвана във клиентското приложение, за да може да клиента да открива на картата най-близките свободни велосипеди. Може да има няколко такива API-та, тъй като в изискването е споменато, че VeloCity трябва да може да се интегрира със всички доставчици на Maps услуги.

3. Описание на допълнителни структури

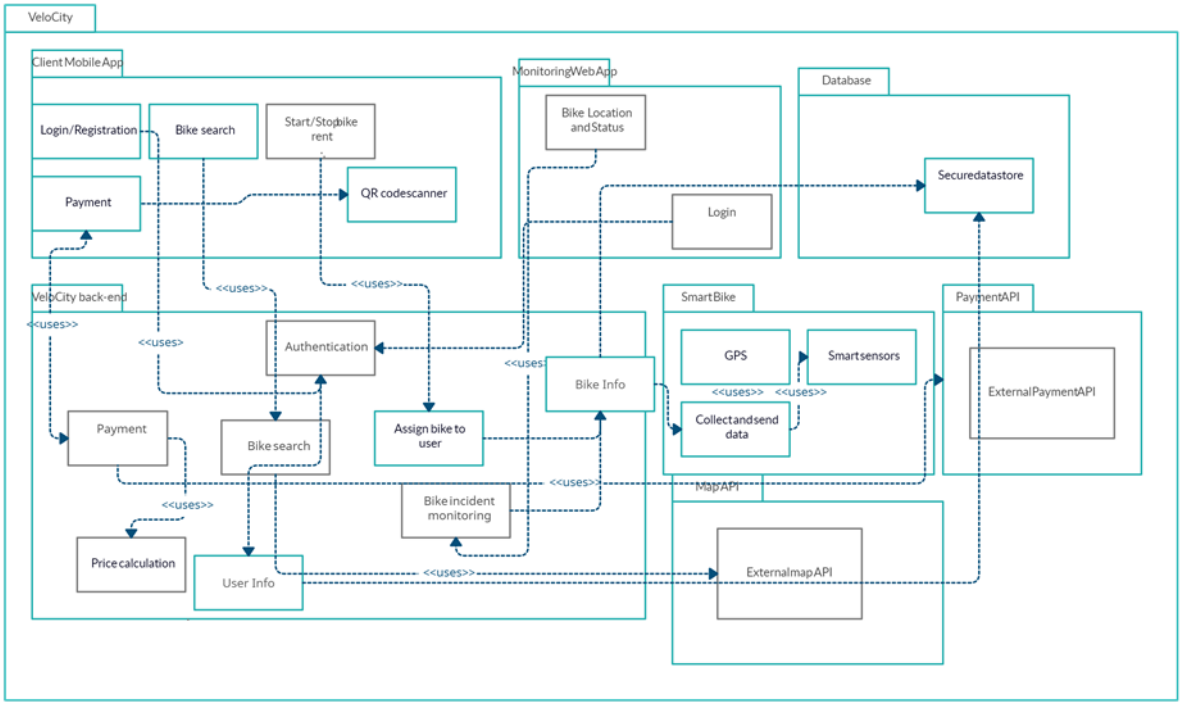
3.1. Структура на употреба на модулите

3.1.1. Мотивация на избор

Първата допълнителна структура която избрахме да опишем е Структура на употреба на модулите. В нея връзките между модулите са от вида “X използва Y”. Структурата на употреба е много подходяща за системи изградени от подмодули които активно комуникират помежду си. VeloCity се състои база от данни, два клиента и един сървър, два външин интерфейса, както и умно колело оборудвано със сензори. Те активно комуникират един с друг, което прави тази структура подходяща за по-доброто разбиране на архитектурата на VeloCity. С тази структура могат да се определят необходимите интерфейси които всеки модул трябва да преодставя, както и вида данни които трябва да се обменят.

Със тази структура, също така, може да се определят кои модули в системата са зависими един от друг и кои не са. По този начин могат да се обособят отделни единици които са независими и могат да бъдат разработвани самостоятелно. Това е мощен инструмент за по-ефективна разбработка на една софтуерна система.

3.1.2. Първично представяне



3.1.3. Описание на елементите и връзките

3.1.3.1. Client Velocity App: Login/Registration -> Velocity back-end: Authentication

При логин и регистрация модулът login/registration в клиента използва Authentication модулът във сървъра. Ако автентикацията е успешна Authentication модулът ще върне подходящо съобщение както и сесиен индентификатор на клиента. Чрез него клиента ще може да бъде оторизиран за извършване на различни действия в приложението.

3.1.3.2. Client Velocity App: Bike Search -> Velocity back-end: Bike Search

Bike Search модулът в Velocity App използва Bike Search модулът от сървъра. Сървърът връща информация за най-близките велосипеди, които се визуализират от клиента чрез видима карта.

3.1.3.3. Velocity back-end : User info -> Secure Data Store

User Info модулът изпозлва базата от данни за да записва информация за потребителите, и за да може да достъпва информация от базата данни за потребителите. Тази инфромация включва ЕГН, име парола имейл и други.

3.1.3.4. Velocity back-end: Bike Info -> Smart Bike: Collect and Send Data

Bike Info модулът достъпва информация за състоянието на велосипедите като например местоположение, зареденост на батерия и данни от останалите смарт сензори .Тази информация е предоставена от Collect and Send Data модулът.

3.1.3.5. Velocity back-end: Bike Info -> Database: Secure Data Store

Bike Info модулът използва базата от данни за да записва състоянието на велосипедите, като например дали велосипедът е зает, данните от сензорите както и местоположението на велосипеда. Също така Bike Info използва базата от данни за да може да достъпва същите тези данни които впоследствие могат да бъдат предоставени на други модули.

3.1.3.6. Velocity back-end: Authentication -> Velocity back-end: User Info

Authentication модулът изпозлва потребителските данни преодставени от User Info (например парола) за да провери дали автентикаията е успешна.

3.1.3.7. Velocity back-end: Payment -> Velocity back-end: User Info

APayment модулът изпозлва Price Calculation модулът за да може да начисли необходимата сума за плащане.

3.1.3.8. Velocity back-end : Bike Incident monitoring -> Velocity back-end: Bike Info

PaBike Incident Monitoring модулът използва данните от сензорите на велосипеда за да може да определи чрез

специални digital signal processing алгоритми дали е настъпил инцидент с велосипеда.

3.1.3.9.Smart Bike: Collect and Send Data -> GPS

Collect and send data чете през определен интервал данните за местоположението на колелото предоставени от GPS модулът. Интервала трябва да е под 5 секунди.

3.1.3.10. Smart Bike: Collect and Send Data -> Smart Sensors

Collect and send data чете през определен интервал данните за физическите характериситки на батерията, както и данните от шок сензорите предоставени от Smart Sensors. Интервала трябва да е по-малък от 5 секунди.

3.1.4. Описание на обкръжението

Системата се свързва и използва външни услуги за карти (Google maps, BG maps, Open Street maps и т.н.), както и Външни API за извършване на онлйан плащания.

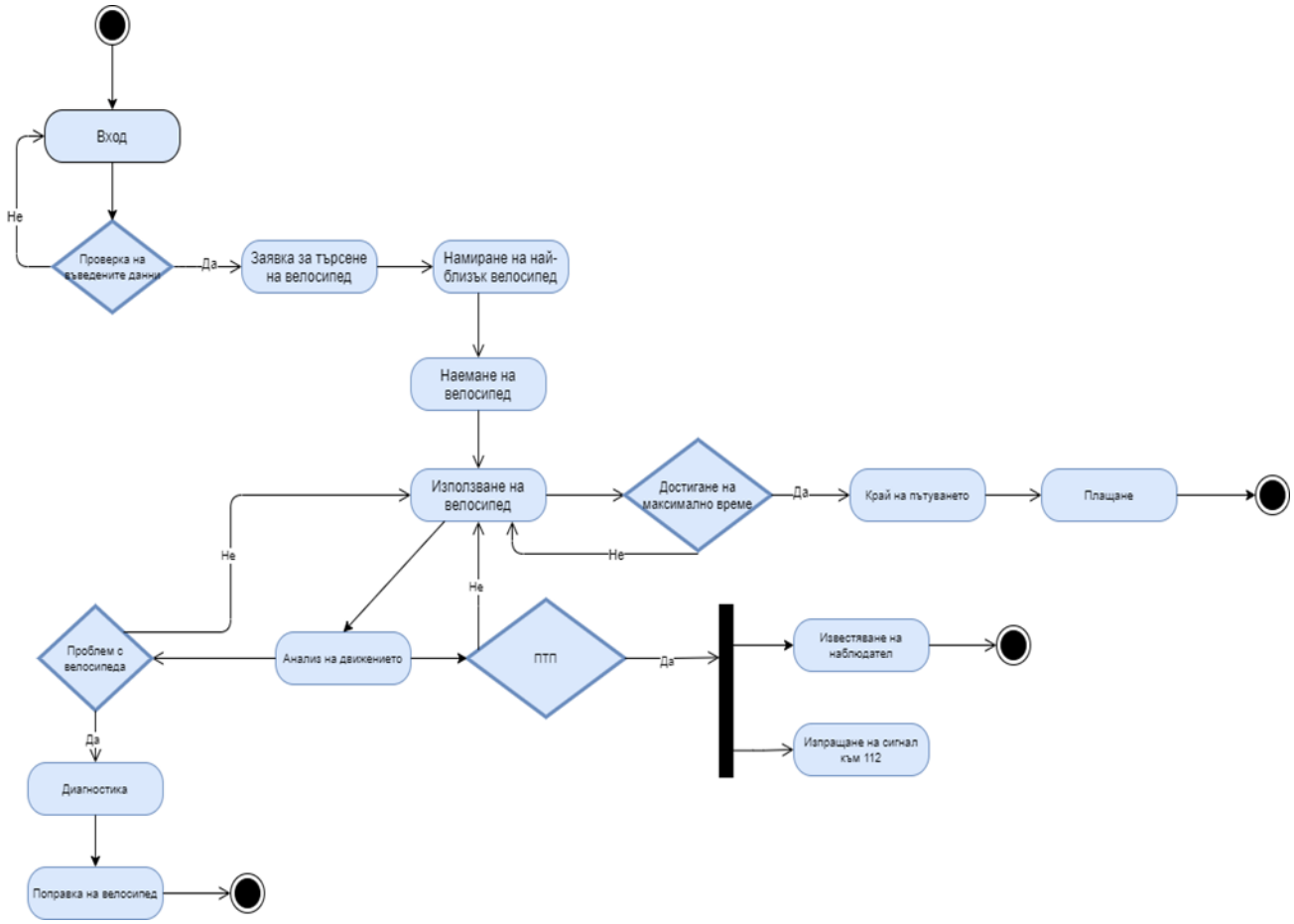
3.2. Структура на процесите

Втората допълнителна структура, която избрахме да опишем е Структура на процесите. В нея елементите са компоненти, които се проявяват по време на изпълнението и средствата за комуникация между процесите.

3.2.1. Мотивация на избор

Структурата показва нагледно функционалността на системата VeloCity. Целта на структурата е да представи последователността по време на изпълнение на процесите, както и как си взаимодействат основните изчислителни процеси. Структурата е полезна при документацията на комплексни системи като VeloCity.Тази структура ще се използва от крайните потребители, дизайнерите на потребителския интерфейс и QA инженерите.

3.2.2. Първично представяне



3.2.3. Описание на елементите и връзките

Първата стъпка след влизането в системата е потребителят да въведе своето потребителско име и парола. Системата проверява автентичността на въведените данни. При грешка, потребителят трябва да въведе наново данните си. При намерено съвпадение на въведената информация с базата от данни, влизането в системата е успешно.
Следващата стъпка е избор на опцията за търсене на близък велосипед. Системата гарантира бързото намиране на най-близък до местоположението на потребителя велосипед отговарящ на изискването да има поне X% батерия.
Следващата стъпка, която трябва да предприеме потребителя е да наеме велосипеда. При избор на тази опция потребителя трябва да отбележи велосипеда, който иска да наеме и да избере начин на плащане. При успешното преминаване на тази стъпка се активира велосипеда и е в процес на използване. По време на фазата на използване на велосипеда е активен процеса на Анализ на движението. Той отговаря за безопасното използване на велосипедите като проверява за евентуално

настъпило ПТП или за проблем с велосипеда. При настъпило ПТП се изпраща сигнал към 112 , както и на наблюдател. При настъпила авария с велосипеда се прави диагностика на проблема и се изпраща за поправка.
При изтичане на максималното време за ползване на велосипеда се прекратява сесията на използване и се преминава към финалната стъпка – Плащане.

3.2.4. Описание на обкръжението

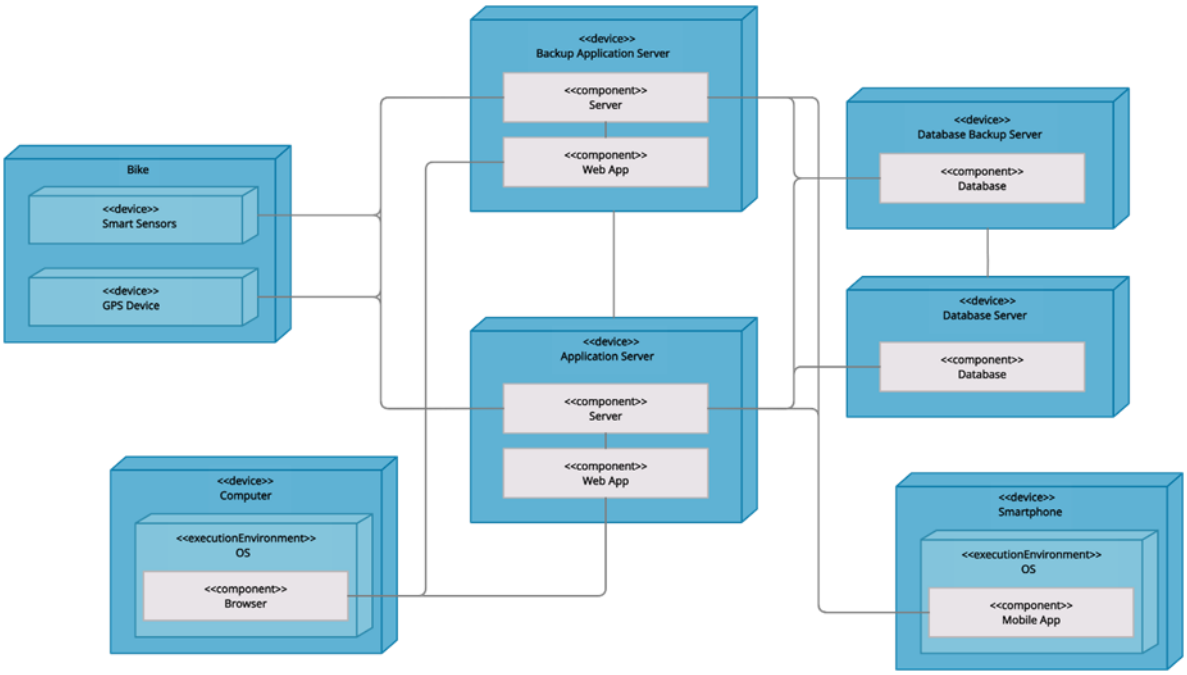
Регистрацията се извършва през Интернет, като преминава през различни валидации и проверки.При използването на GPS е възможно използването на карти като Google maps, Waze и Maps.

3.3. Структура на внедряването

3.3.1. Мотивация на избор

Една от допълнителните структури е структурата на внедряването, за да се покаже разположението на софтуера върху хардуера и какви са комуникационните връзки между устройствата. Системата има клиенти, сензори на велосипедите, сървър и база данни, за които е важно да бъде уточнено как се внедряват. Клиентите използват системата през мобилно уеб приложение, което се свързва със сървъра с помощта на HTTP протокол. Също така всеки велосипед е снабден с умни сензори, които отчитат различни параметри и модулът, който се грижи за събирането на данните и обработката им в такъв формат, че те да могат да бъдат изпратени до сървъра чрез HTTP протокол. Сървърът на системата съдържа логиката за функционалностите на системата. Има и сървър, на който се съхраняват данните в системата. С помощта на структурата на внедряването организацията на модулите по време на разработка става по-добра и се улеснява процеса на разпределение на работата по екипи, защото системата освен че е разделена на модули, тя е ясно разделена и на хардуерно ниво. Част от определянето на какви и колко големи хардуерни ресурси трябва да бъдат на разположение за реализирането на системата е изготвянето на тази структура. Тези ресурси са съществени при определянето на себестойността на разработването и поддръжката на системата.

3.3.2. Първично представяне



3.3.3. Описание на елементите и връзките

На първоначалното представяне структурата на внедряване сме представили как различните софтуерни компоненти са разположени върху хардуера. Също така сме показали и комуникационните канали между отделните елементи на системата.

- Application Server – това е сървърът, на който е разположено самото приложение, тоест логиката на системата. В него сме представили два компонента:
    - Server – на този компонент е разположена логиката на системата. Тук се намира модулът Client Mobile App.
    - Web App – този компонент отговаря за уеб частта на системата. Тук са разположени модулите VeloCity Back-end и Monitoring Web App.

- Application Server има свой дубликат, както се вижда и от първичното представяне на структурата (Backup Application Server). Този резервен сървър отговаря за осигуряването на по-добра наличност и производителност при големи натоварености на

системата или при срив на основния сървър. Двата сървъра работят на принципа на пасивен излишък, като главния сървър отговаря за синхронизацията на състоянието на резервния. Затова има и връзка между двата сървъра, чрез която да се осъществи самата синхронизация.

- Database Server - това е сървърът, на който се съхранява цялата база от данни на системата. Тъй като не може да си позволим възможността за изгубване на цялата база от данни или неналичност на системата при някакъв проблем с Database Server, имаме Database Backup Server, който отново работи на принципа на пасивен излишък и главния сървър отговаря за синхронизацията на състоянието на резервния. Application Server комуникира с основния сървър за базата от данни Database Server, освен ако няма някакъв проблем с него. Ако има проблем с основния сървър за базата от данни, тогава Application Server се обръща към Database Backup Server.

- Computer – това е устройството, което се използва за да могат администраторите, наблюдателите и механиците да влязат в уеб приложението на системата с помощта на браузър.

- Smartphone – това е устройството, с което обикновеният потребител на системата я използва. На смартфона потребителя трябва да има съответното мобилно приложение.

- Bike – това е велосипедът, който е снабден с устройствата Smart Sensors и GPS Device.

    - Smart Sensors – тези устройства събират и дават информация за състоянието на велосипеда и околната среда.
    - GPS Device – дава информация за местоположението на велосипеда.

3.3.4. Описание на обкръжението

В структурата на внедряване е наблегнато на разположението на системата, а не толкова на обкръжението ѝ, затова не са показани всички връзки на системата ни с външни за нея системи. Въпреки това е представена базата от данни, която е

външна за системата и се намира на отделен сървър, тъй като нейното разположение е важно да бъде показано, както и да се изясни как ще се случва комуникацията с нея.

3.3.5. Описание на възможни вариации

В случай, че системата започне да се пренатоварва често, поради голям брой клиенти, е възможно да се добавят нови дубликати на Application Server, които да поемат част от работата и така да се предотврати пренатоварването. Същото важи и за Database Server. При необходимост може да се добавят още негови дубликати, ако Database Backup Server не е достатъчен, за насмогване със заявките към базата от данни. И при двете възможни вариации трябва да се осигури синхронизацията между дубликатите, иначе ще се появят проблеми в системата.

3.3.6. Архитектурна обосновка

Със структурата на внедряването целим да покажем физическото разположение на системата ни, тъй като то е много важно за всяка система. Избрахме тази структура, защото системата ни ще трябва да може да работи едновременно с много потребители, а затова е необходима съответната хардуерна поддръжка, чрез която да осигурим обработката на много заявки от различни потребители. Тази структура също ще гарантира и по-добра наличност на системата, което пък от своя страна ще допринесе за използваемостта й, защото ако има чести откази и системата е неналична, то тя на практика става неизползваема за потребителите.
Структурата може да послужи на заинтересованите лица в различна степен, но най-вече ще послужи на различните анализатори на системата да осигурят изисканите качества за системата, както и на екипа за внедряване и администраторите на системата.

3.3.7. Допълнителна информация

Използван е курсивен шрифт за рефериране на различните модули на системата, които са описани в структурата декомпозиция на модули.

4. Архитектурна обосновка

Архитектурните драйвери (основните изисквания на системата) и обосновка защо са избрани:

4.1. Функционални

4.1.1. Поддръжка на следните групи потребители: наемател на велосипед, техник, системен администратор и наблюдател.

Наличието на четирите различни роли определя голяма част от моделираните обекти, както и дава поглед върху различните сценарии на употреба. Поради това смятаме, че е архитектурен драйвер.

4.1.2. Уеб приложение за системните администратори и наблюдателите.

Архитектурен драйвер, защото ни дава информация за съществуването на още един модул освен мобилното приложение - а именно уеб приложение. Чрез него администраторите трябва да могат да извършват технически проверки, а наблюдателите да следят различните показатели на велосипедите и да сигнализират при нередности. Това ни показва детайли за няколко сценария на употреба. Също така разбираме какъв начина, по който ще се достъпва - чрез web browser.

4.1.3. При търсене на велосипед се намира такъв с поне X% батерия.

Възможността за намиране на близки велосипеди е архитектурен драйвер. Това е основна функционалност без която трудно би се реализирала основната идея на приложението. Откриването на велосипедите в градската среда трябва да бъде максимално улеснено. Карта показваща най-близкия велосипед, който би свършил работа на потребителя би улеснила това значително.

4.1.4. Заплащане на услугата чрез кредитна карта, СМС или чрез въвеждане на уникален код от закупен талон.

Възможността за плащане е архитектурен драйвер, защото е основна бизнес функционалност. Без да може да се извършват плащания продукта ни няма да има бизнес стойност. Добре е да има разнообразие от методи на плащане, но като за начало е задължително поне един от тях да присъства. Плащането ще бъде реализирано чрез използване на външна система за

онлайн плащания, която е вече утвърдена. Това ни показва връзка към външен модул.

4.1.5. Изпращане на данни за състоянието на велосипеда и известия при проблем до групите по техническа поддръжка.

Архитектурен драйвер, защото ни показва че всеки велосипед трябва да може изпраща данни към сървъра на системата. Това ни показва връзката между два модула в системата. От една страна имаме велосипеда оборудван със смарт сензори от друга имаме сървъра на приложението, който получава данни от сензорите. Трябва да се проектира система, която да може да събира данните от сензорите и да ги изпраща към сървъра. Това обособява отделен модул, мястото на внедряване, както и вида на данни, които ще се обменят между него и сървъра на приложението.

4.1.6. След изтичане на максималното време наемателят се известява, за да остави велосипеда.

Архитектурен драйвер, защото ни показва връзка между два модула. Трябва да проектираме така архитектурата, че сървъра да може да изпраща проактивно съобщения към мобилното приложение. Това означава използването на архитектурен стил или тактика, която позволява тази функционалност.

4.2. Нефункционални(качествени)

4.2.1. Защита на данните.

Освен посочените основополагащи функционалности, които системата трябва да предоставя, се налага и съобразяването с някои изисквания за качество. Първоначално при регистрацията на потребителите ще се изисква от тях да напишат имената си, ЕГН-то си, телефон за връзка и адрес, за да може наблюдателите да се свържат с тях при възникването на някакъв проблем. Втората стъпка за неприкосновеност на данните е тяхната защита от изгубване и неоторизиран достъп. Наличието на лична информация на потребителите предизвиква предприемане на строги мерки за правилното й съхранение чрез използването на надеждни системи за сигурност и евентуално да се съхранява копие на информацията на друг сървър.

4.2.2. Отказоустойчивост на системата.

Приложението ще бъде такова, че ще бъде част от ежедневието на потребителите. В пиковите часове, когато трябва да се пътува за работа или нещо подобно, то ще бъде интензивно използвано от много различни потребители. За тази цел трябва да се увеличи максимално надеждността на системата така, че това да не създава осезаемо усещане за забавяне или да предизвика сривове, които могат да предизвикат външен достъп до личните данни на потребителите, както и възможност за изчезване на някой от велосипедите.