Курсов проект

по Софтуерни архитектури и разработка на софтуер

на тема

Валерия Кирилова, 62305
Елиана Щерева, 62249
Станимир Петров, 62285

1. Въведение

1.1. Обща информация за текущия документ

1.1.1. Предназначение на документа

Този документ има за цел да представи архитектурата на нашата система Drones Park. Представили сме различни видове структури на архитектурата, с цел по-голяма яснота.

1.1.2. Описание на използваните структури на архитектурата.

1.1.2.1. Декомпозиция на модулите

Декомпозиция на модулите е поглед към реализацията на системата, а не към изпълнението ѝ и е подходяща структура за разпределението на работата по екипи. Представят се различните модули на архитектурата, логическата свързаност между тях, както коя функционалност в кой модул се намира. Основните модули, които сме включили в нашата система са - App, Server и Database config. Тъй като са възможни бъдещи изменения в системата, този архитектурен стил ще допринесе за бързата промяна, както и за последователността по време на разработката.

1.1.2.2. Допълнителни структури

1.1.2.2.1. Структура на процесите

Структурата на процесите изобразява някои функционалности, които предоставя нашата система DronesPark. В нея са показани процесите, които се изпълняват, и последователността от действия на системата, които са необходими да бъдат извършени, за да се предостави дадената функционалност. С цел по-голяма нагледност сме обособили в кои части от системата се извършва всеки един процес и сме показали как се извършват процесите по етапи. Представили сме и различните сценарии, които могат да възникнат при изпълнението на процесите. Този подход допринася по-лесно да се разбере как работи системата.

- Диаграма, показваща процесите, които се извършват при подаден сигнал за нарушение
- Диаграма, показваща процесите, които се извършват при повреда на дрон
- Диаграма, показваща процесите, които се извършват, когато потребител иска да се абонира за паркомясто

1.1.2.2.2. Структура на внедряването

Тази структура представя върху какво са разположени изпълнимите файлове на нашата система и как си комуникират отделните компоненти на системата и елементите на околната среда. Показва какви хардуерни компоненти ще са включени в нашата система, също и колко инстанции на сървъра и базата данни се предвиждат. Подходяща е за представяне на някои от тактиките, предвидени в архитектурата. Позволява да се разберат особеностите относно бързодействието, производителността, отказоустойчивостта, интегритета на данните, надеждността, сигурността и т.н.

1.1.2.2.3. Многослойна структура

Многослойната структура е свързана с това отделните модули да са обособени в отделни слоеве, които могат да си комуникират само със следващия слой. Така ако са в отделни слоеве при евентуален отказ или добавяне на нова функционалност в някой слой, това няма да доведе до сриване на цялата система, а може да се усети минимално само в следващият слой или въобще да не се усети. Има много ползи от използването на многослойна

архитектура, включително мащабируемост, производителност и наличност. Тази структура е подходяща за нашата система, защото е вероятно да настъпят изменения в нея.

1.1.3. Структура на документа

Документът се състои от 5 секции:

- Първа секция - Въведение
- Втора секция - Архитектурна структура “Декомпозиция на модулите”
- Трета секция - Допълнителни архитектурни структури “Структура на процесите”, “Структура на внедряването” и “Структура на слоевете”
- Четвърта секция - Описание на архитектурата

1.2. Общи сведения за системата

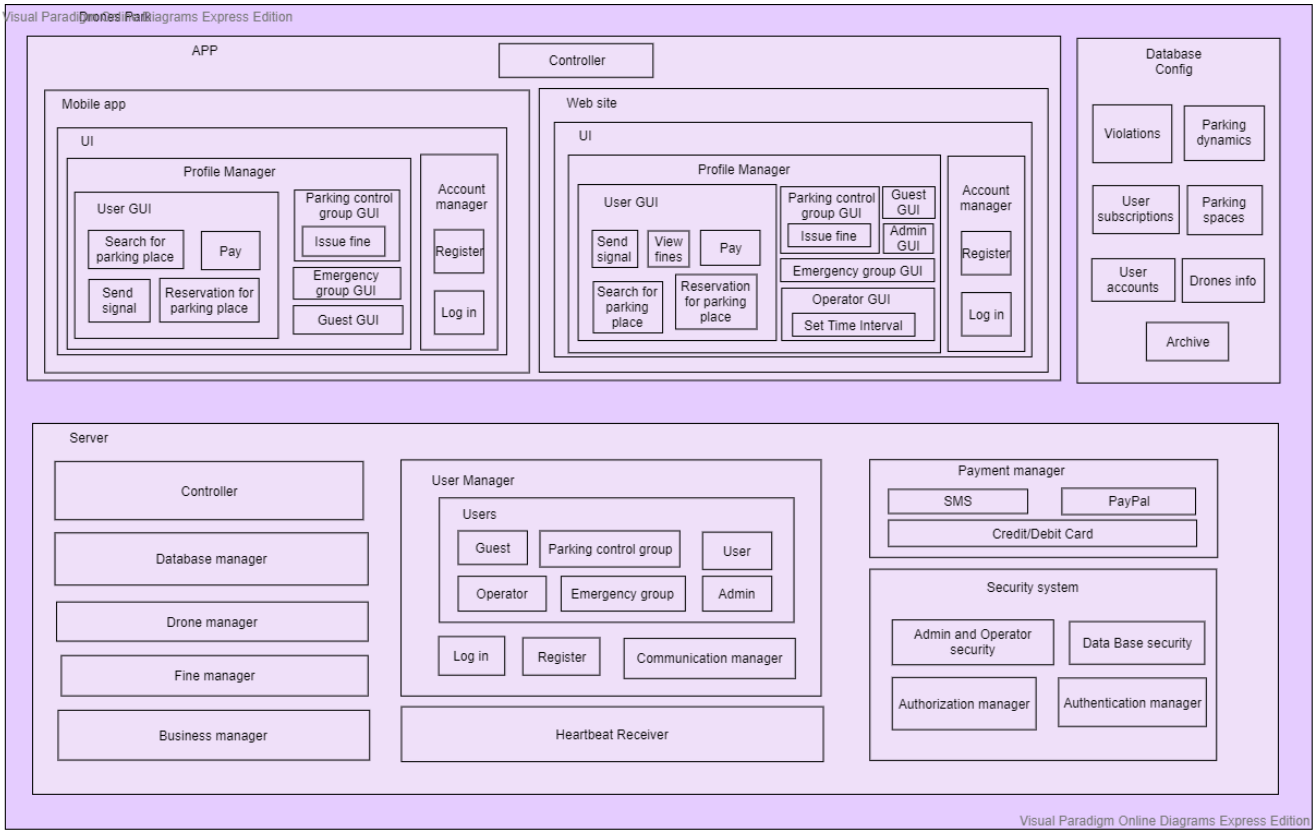
DronesPark е система за следене и управление на свободните паркоместа чрез система от дронове. Целта на нашата система е да осигури места за паркиране с помощта на система от дронове, които заснемат зоните за паркиране, а чрез специфичен алгоритъм сървърът идентифицира свободните паркоместа. Системата може да бъде използвана от всеки, който иска да намери удобно за него паркомясто бързо и лесно. Регистрираните потребители имат право да избират срока на абонамента, както и различни начини за плащане. Системата поддържа следните групи потребители - Администратор, Оператор, Аварийни групи, Групи по контрол на паркирането (т.нар. „паяци“), Регистрирани потребители, както и Обикновени потребители (Гости). Подсигурили сме отделни модули, които да се грижат за производителността, надеждността, отказоустойчивостта и изправността на нашата система.

1.3. Терминологичен речник

- **Heartbeat receiver** - даден компонент периодично излъчва сигнал, който друг компонент очаква. Ако сигналът не се получи, се предполага, че в компонент А е настъпила повреда и се задейства процедура за отстраняване на повредата. Сигналът може да носи и полезни данни – например, както в нашата система, дронът може да изпраща локация на даден сървър.
- **Weather monitoring system** - Метеорологична станция е съоръжение, което включва в себе си сензори за измерване на атмосферните условия. Предоставя информация на база измервания на температура , атмосферно налягане, влажност, трайно намалена видимост, скорост на вятъра и посока на вятъра.
- Използвани протоколи за комуникация със сървъра:
  - **HTTPS** - Hypertext Transfer Protocol Secure
  - **JDBC** - Java Database Connectivity
  - **EBICS** - Electronic Banking Internet Communication Standard
  - **TCP/IP** - Internet protocol suite
  - **SSL/TLS** - Secure Sockets Layer

2. Декомпозиция на модулите

2.1. Общ вид на декомпозицията на модули за системата

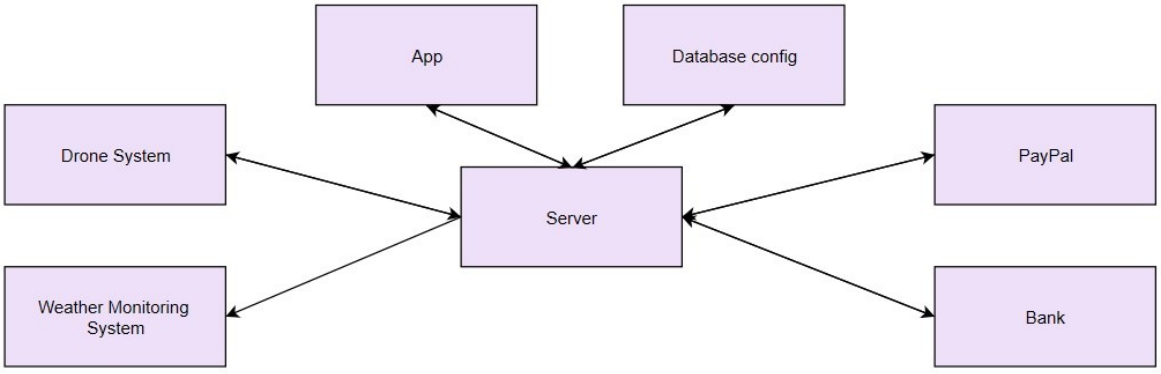


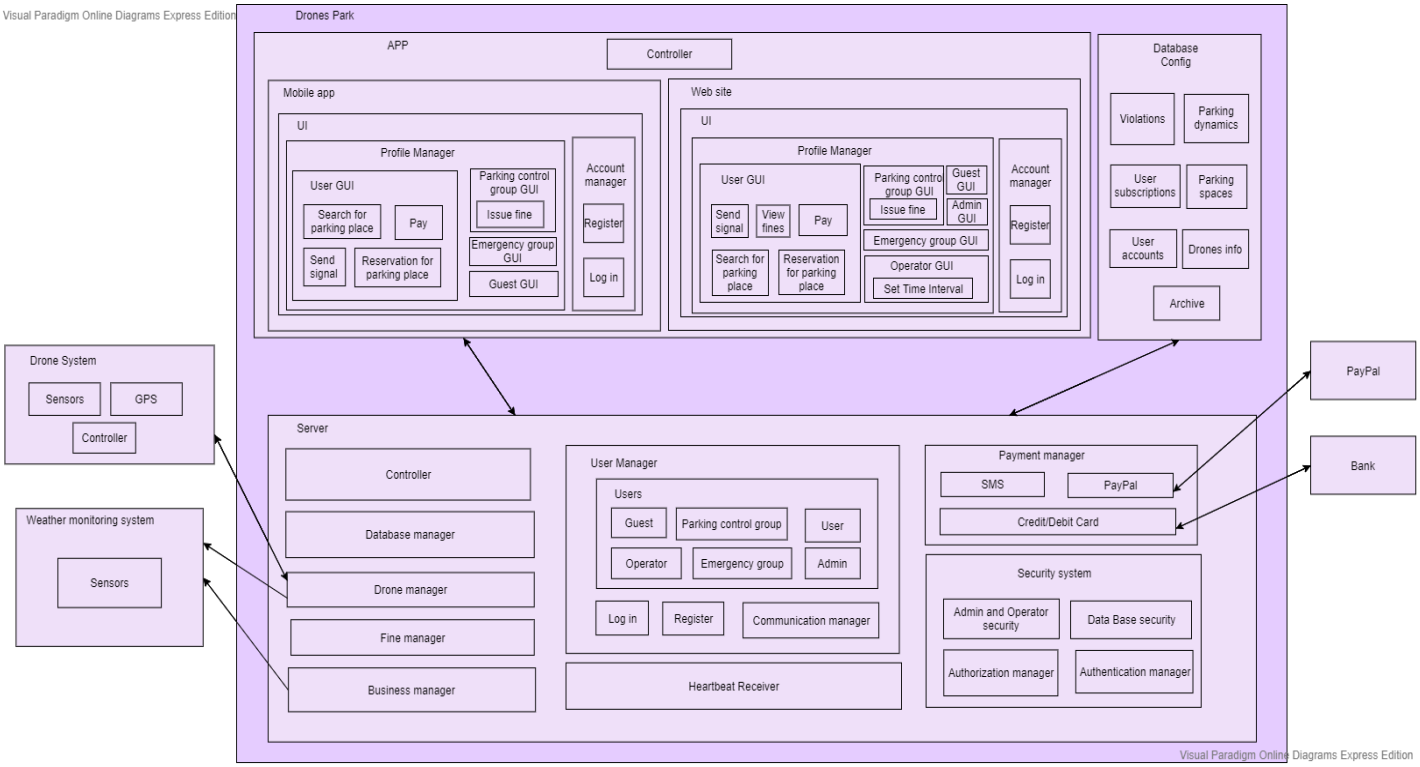
Нашата система включва три основни модула:

- App - модул за връзка с клиентите на системата, чрез мобилно приложение или уеб сайт.
- Server - съдържа основната бизнес логика на системата.
- Database config - дефинира конфигурационни параметри на използваните файлове, които ще се съхраняват в нашата база данни.

Потокът на данни преминава от Database config през сървър и достига до App.

2.2. Контекстна диаграма

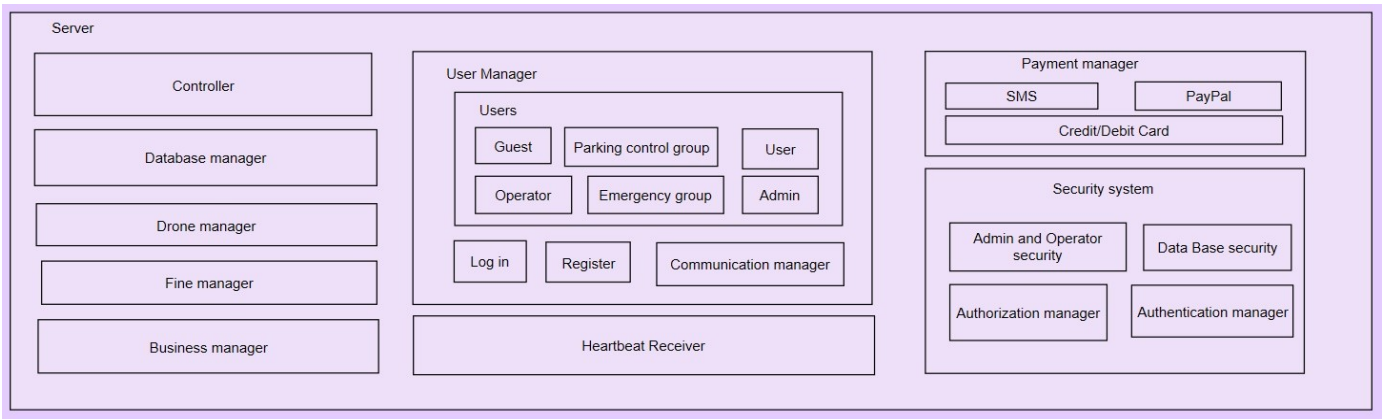




- App - за комуникация със сървъра се използва подмодула Controller.
- Server - комуникира си с външни системи Drone System, Weather monitoring system, PayPal и Bank чрез протоколи. За връзка с модулите App и Database config използваме подмодулите съответно User manager и Database manager (чрез HTTPS протокол).
- Database config - комуникира с Server чрез HTTPS.

2.3. Подробно описание на всеки модул

2.3.1. Server



2.3.1.1. Предназначение на модула

Този модул е предназначен, както да осъществява комуникация между отделните модули, така и за реализацията на бизнес логиката на системата.

2.3.1.2. Основни отговорности на модула в системата

Този модул е връзката на системата с външните системи. Служи за реализацията на функционалностите в системата, както и да осъществява комуникацията между потребителите и базата данни. Също така се грижи за проверката и осигуряването на защита на данните.

2.3.1.3. Описание на интерфейсите на модула.

2.3.1.3.1. Controller

Приема заявки от UI и ги изпраща на съответния модул.

2.3.1.3.2. Database manager

Database manager-а се грижи за обработване на заявките към базата данни.

2.3.1.3.3. Fine manager

При издаване на фиш, Fine manager-а записва фиша в базата данни. При поискване на информация за даден фиш се свързва с Database manager, който му връща информация.

2.3.1.3.4. Business manager

Бизнес мениджърът определя цената за паркоместата, за които няма абонамент, в определен ден и час на база предходни предвиждания за честотата на заемане на съответното паркомясто, както и от предоставени данни за времето от метеорологичната система.

2.3.1.3.5. User manager

User manager се грижи за регистрацията на потребителите, за определянето на права на съответните групи потребители. Също така при вход на потребител изпраща заявка до Authentication мениджъра, за да бъде идентифициран потребителят.

2.3.1.3.5.1. Users

Описва различните видове потребители на системата и техните права.

2.3.1.3.5.1.1. Guest

2.3.1.3.5.1.2. Parking control group

Групите по контрол приемат сигнали и следят за нарушители. При засичане на нарушител, освен принудителното преместване на автомобила, заснемат настъпилото събитие и издават фиш.

2.3.1.3.5.1.3. User

Регистрираните потребители могат да търсят, да правят резервация и да заплащат абонамент за определено паркомясто за определен период, да подават сигнали за нарушители. Също така през уеб сайт могат да прегледат издадените им фишове.

2.3.1.3.5.1.4. Operator

Получава сигнал при трайно намалена видимост от метеорологичната система. Задава интервал от време, на който да се обновява информацията за свободните места.

2.3.1.3.5.1.5. Emergency group

Получават сигнали от системата за повредени дронове и ги поправят.

2.3.1.3.5.2. Log in

Потребителят въвежда данните в UI, който ги изпраща на контролера, а той се свързва с User manager за проверка на входните данни с тези от БД, както и проверка от Authentication и Authorization в системата за сигурност.

bool Login(string username, string password);

- Вход: потребителско име; парола
- Изход: съобщение дали влизането е успешно

2.3.1.3.5.3. Register

Потребителят въвежда данните в UI, който ги изпраща на контролера, а той се свързва с User manager за валидиране на входните данни. При успешна валидация те се записват в базата

данни. При регистрация User manager валидира данните и ги изпраща за съхранение в базата данни.

bool RegisterUser(string username, string password, string email);

- Вход: потребителско име; парола; имейл
- Изход: съобщение дали е била успешна регистрацията

2.3.1.3.5.4. Communication manager

Осъществява комуникацията между отделните групи потребители, както и изпраща съобщения от системата към потребителите.

2.3.1.3.6. Security system

С цел по-висока защита използваме автентикация, оторизация, както и криптиране на данните в БД.

2.3.1.3.6.1. Admin and operator security

С цел 100% защита на потребителите от тези групи, техният достъп до системата ще е възможен само от определени устройства и IP адреси.

2.3.1.3.6.2. Database security

Осигурява конфиденциалност на данните посредством криптиране.

2.3.1.3.6.3. Authorization manager

Проверява дали потребителят има достъп до определени ресурси.

2.3.1.3.6.4. Authentication manager

При получаване на заявка от UI за вход или регистрация, контролерът я обработва и изпраща на Authentication manager, който проверява дали потребителят е този, за когото се представя.

2.3.1.3.7. Payment manager

Payment manager следи за извършване на плащания в UI от регистрирани потребители, свързва се с външни системи и връща резултат дали е било успешно. Ако резултатът е бил положителен се записва в БД.

2.3.1.3.7.1. PayPal

Препраща потребителя към външна система PayPal.

2.3.1.3.7.2. SMS

Показва информация за плащане чрез SMS. След това потребителя има 5 min да изпрати SMS на посочения номер с посоченото съдържание. Ако такъв не бъде получен, плащането се счита за неуспешно и процесът на плащане трябва да се започне отначало.

2.3.1.3.7.3. Credit/debit card

Препраща потребителя към външна услуга за плащане с карта.

2.3.1.3.8. Heartbeat receiver

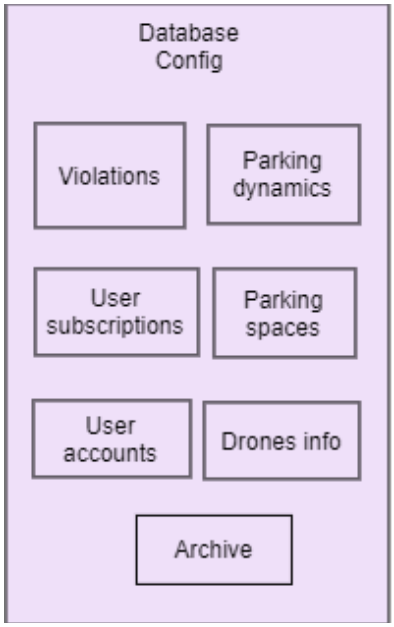
Дроновете периодично излъчват сигнал към системата. Heartbeat receiver следи за получаването на този сигнал и ако сигналът не се получи най-вероятно някой дрон не функционира правилно. В такъв случай аварийните групи биват уведомени.

2.3.1.3.9. Drone manager

Определя броя на летящите в момента дронове и маршрута на всеки от тях, използвайки данни от предходни дни за динамиката на паркирането

и данни от метеорологичната система (при получаване на информация за трайно намалена видимост се уведомява операторът на системата). На базата на тази информация прави статистика, която се записва в базата данни. Чрез специфичен алгоритъм разпознава свободните паркоместа на база снимките, изпратени от системата от дронове.

2.3.2. Database Config



2.3.2.1. Предназначение на модула

Database config съдържа информацията, необходима за взаимодействието (създаване, четене и промяна) с базата данни. Този модул пази стойности за различни конфигурационни параметри, които влияят върху употребата на базата данни, като например:

- параметри, необходими за изграждането на базата данни
- параметри показващи текущото състояние на базата данни
- параметри, определящи количеството системни ресурси, които дадена операция от базата данни може да използва

2.3.2.2. Основни отговорности на модула в системата

Модулът съдържа конфигурационни файлове, определящи съдържанието на базата данни. В базата данни ще се съхранява информация относно потребителите на нашата система, техните абонаменти, нарушения, също така и информация свързана паркоместата, динамиката на паркиране в зоните, както и информация свързана с дроновете.

2.3.2.3. Описание на интерфейсите на модула.

Задават се параметри за следните таблици в базата данни:

2.3.2.3.1. Violations

Информация за извършени нарушения, подадени от групите за контрол (снимка на нарушението, електронен фиш). Тази информация ще бъде достъпна през публичен уебсайт.

2.3.2.3.2. Parking spaces

Информация за всички места за паркиране (дали дадено място е свободно или има платен абонамент).

2.3.2.3.3. Drones info

Информация за последно известно местоположение на всеки от дроновете. Тази информация е необходима за определяне на

предполагаем район, за да се уведомят аварийните групи, ако някой дрон излезе от строя.

2.3.2.3.4. Archive

Системата поддържа архив на данните за динамиката на паркирането и всички издадени фишове за глоби за 25 години назад във времето, както и архив на заснетите изображения за 3 години назад.

2.3.2.3.5. User accounts

Съдържа информация за потребителските акаунти (потребителско име, парола, e-mail).

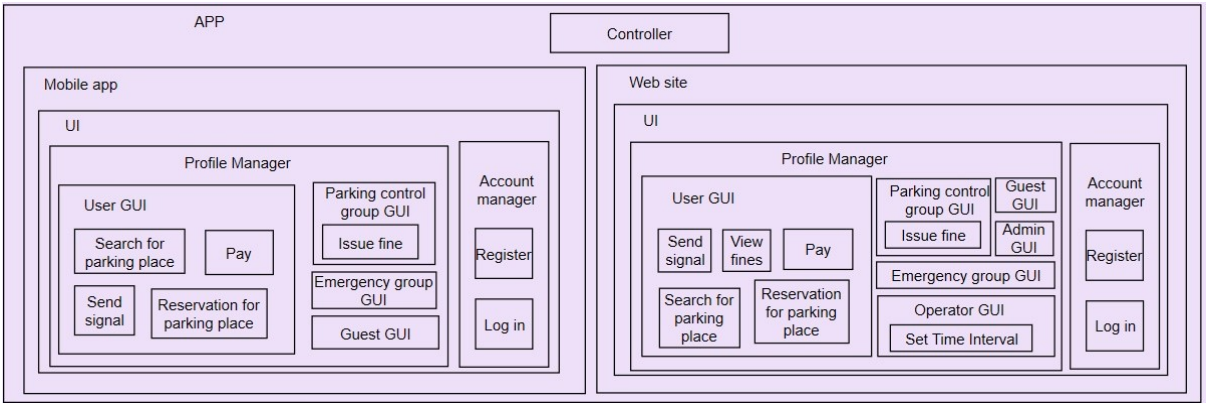
2.3.2.3.6. Parking dynamics

Тук пазим данни от предишни дни за честотата на заемане/освобождаване на места спрямо съответния час и ден от седмицата.

2.3.2.3.7. User subscriptions

Тук пазим данни за активните абонаменти за паркиране.

2.3.3. App



2.3.3.1. Предназначение на модула

Този модул ни предоставя мобилно приложение и уеб сайт, които потребителите на системата използват.

2.3.3.2. Основни отговорности на модула в системата

Модулът служи за комуникация между потребителите и сървъра. Предоставя по подходящ начин данните, така че те да бъдат разбрани. Предназначен е, за да улесни използването на системата и нейните функционалности.

2.3.3.3. Описание на интерфейсите на модула

2.3.3.3.1. Controller

Грижи се за обработване на заявките от и към сървъра.

2.3.3.3.2. Web site

Използването на функционалностите на системата е възможно чрез публичен сайт. През него са достъпни снимката и фишът, които са издадени от групите по контрол.

2.3.3.3.2.1. UI

Всички елементи, с които потребителят си взаимодейства. Подпомага за по-лесното използване на системата, като предоставя различни бутони за различните функционалности, спрямо правата на различните групи потребители

2.3.3.3.2.1.1. Account manager

2.3.3.3.2.1.1.1. Log in

Предоставя шаблон на потребителя за попълване на необходимите данни и получава отговор от системата дали входът е успешен.

bool Login(string username, string password);

- Вход: потребителско име; парола
- Изход: съобщение дали влизането е успешно

2.3.3.3.2.1.1.2. Register

Предоставя шаблон на потребителя за попълване на необходимите данни и получава отговор от системата дали регистрацията е успешна.

bool RegisterUser(string username, string password, string email);

- Вход: потребителско име; парола; имейл
- Изход: съобщение дали е била успешна регистрацията

2.3.3.3.2.1.2. Profile manager

2.3.3.3.2.1.2.1. Guest GUI

2.3.3.3.2.1.2.2. Admin GUI

2.3.3.3.2.1.2.3. Emergency group GUI

2.3.3.3.2.1.2.4. Operator GUI

2.3.3.3.2.1.2.4.1. Set Time Interval

Информацията за свободните места се обновява на определен интервал от време, който се задава от оператора на системата.

2.3.3.3.2.1.2.5. Parking control group GUI

2.3.3.3.2.1.2.5.1. Issue fine

При засичане на нарушители, групите по контрол на паркиране издават електронен фиш, посредством шаблон, който трябва да попълнят, както и да прикачат снимка на нарушителя. След това фишът се изпраща до сървъра за обработка и запис в базата данни.

2.3.3.3.2.1.2.6. User GUI

2.3.3.3.2.1.2.6.1. Pay

Функционалност, която се предоставя само на регистрирани потребители за плащане на абонамент при резервация на свободно паркомясто.

bool Pay(int PaymentMethodId, ParkingPlace place, double price);

- Вход: метод на плащане; паркомясто, за което се плаща; цена на абонамента

- Изход: съобщение дали пращането е извършено успешно
- Ограничения при употреба: PaymentMethodId трябва да бъде валидно ID на някой от предлаганите методи за плащане

2.3.3.3.2.1.2.6.2. Reservation for parking places

Само на регистрираните потребители се предоставя възможност за резервация на паркомясто.

2.3.3.3.2.1.2.6.3. Search for parking places

Предоставя се възможност за търсене на свободни паркоместа, за да може потребителят да си избере най-удобното за него.

List<ParkingPlace> SearchForParkingPlace(Location location, DateTime Start, DateTime End);

- Вход: локация - в която да се търси паркомясто; дата и час на заемане на паркомясто; дата и час на освобождаване
- Изход: списък от паркоместа, отговарящи на условията, зададени от входните параметри

2.3.3.3.2.1.2.6.4. View fines

Предоставя се възможност за преглед на всички фишове, издавани някога за съответния потребител. Тази функционалност е достъпна само през сайта.

2.3.3.3.2.1.2.6.5. Send signal

При установяване на нарушител (неправомерно паркиране, неплатен абонамент), потребителите могат да подават сигнали към системата, за да се вземат необходимите мерки.

bool SendSignal(Location location, string CarRegistrationNumber);

- Вход: локация; регистрационен номер на нарушителя
- Изход: съобщение дали сигналът е успешно приет

2.3.3.3.3. Mobile app

С цел по лесен достъп до системата се предлага и мобилно приложение, което може да се използва от аварийни групи, групи по контрол на паркирането, регистрирани потребители и обикновените потребители.

2.3.3.3.3.1. UI

Всички елементи, с които потребителят си взаимодейства.

2.3.3.3.3.1.1. Account manager

2.3.3.3.3.1.1.1. Log in

Предоставя шаблон на потребителя за попълване на необходимите данни и получава отговор от системата дали входът е успешен.

bool Login(string username, string password);

- Вход: потребителско име; парола
- Изход: съобщение дали влизането е успешно

2.3.3.3.3.1.1.2. Register

Предоставя шаблон на потребителя за попълване на необходимите данни и получава отговор от системата дали регистрацията е успешна.

bool RegisterUser(string username, string password, string email);

- Вход: потребителско име; парола; имейл
- Изход: съобщение дали е била успешна регистрацията

2.3.3.3.3.1.2. Profile manager

2.3.3.3.3.1.2.1. Guest GUI

2.3.3.3.3.1.2.2. Emergency group GUI

2.3.3.3.3.1.2.3. Parking control group GUI

2.3.3.3.3.1.2.3.1. Issue fine

При засичане на нарушители, групите по контрол на паркиране издават електронен фиш, посредством шаблон, който трябва да попълнят. След това фишът се изпраща до сървъра за обработка и запис в базата данни.

2.3.3.3.3.1.2.4. User GUI

2.3.3.3.3.1.2.4.1. Pay

Функционалност, която се предоставя само на регистрирани потребители за плащане на абонамент при резервация на свободно паркомясто.

bool Pay(int PaymentMethodId, ParkingPlace place, double price);

- Вход: метод на плащане; паркомясто, за

което се плаща; цена на абонамента
- Изход: съобщение дали пращането е извършено успешно
- Ограничения при употреба: PaymentMethodId трябва да бъде валидно ID на някой от предлаганите методи за плащане

2.3.3.3.3.1.2.4.2. Reservation for parking places

Само на регистрираните потребители се предоставя възможност за резервация на паркомясто.

2.3.3.3.3.1.2.4.3. Search for parking places

Предоставя се възможност за търсене на свободни паркоместа, за да може потребителят да си избере най-удобното за него.

List<ParkingPlace> SearchForParkingPlace(Location location, DateTime Start, DateTime End);

- Вход: локация - в която да се търси паркомясто; дата и час на заемане на паркомясто; дата и час на освобождаване
- Изход: списък от паркоместа, отговарящи на условията, зададени от входните параметри

2.3.3.3.3.1.2.4.4. Send signal

При установяване на нарушител (неправомерно паркиране, неплатен абонамент..), потребителите могат да подават сигнали към системата, за да се вземат необходимите мерки.

bool SendSignal(Location location, string CarRegistrationNumber);

- Вход: локация; регистрационен номер на нарушителя
- Изход: съобщение дали сигналът е успешно приет

2.4. Описание на възможните вариации

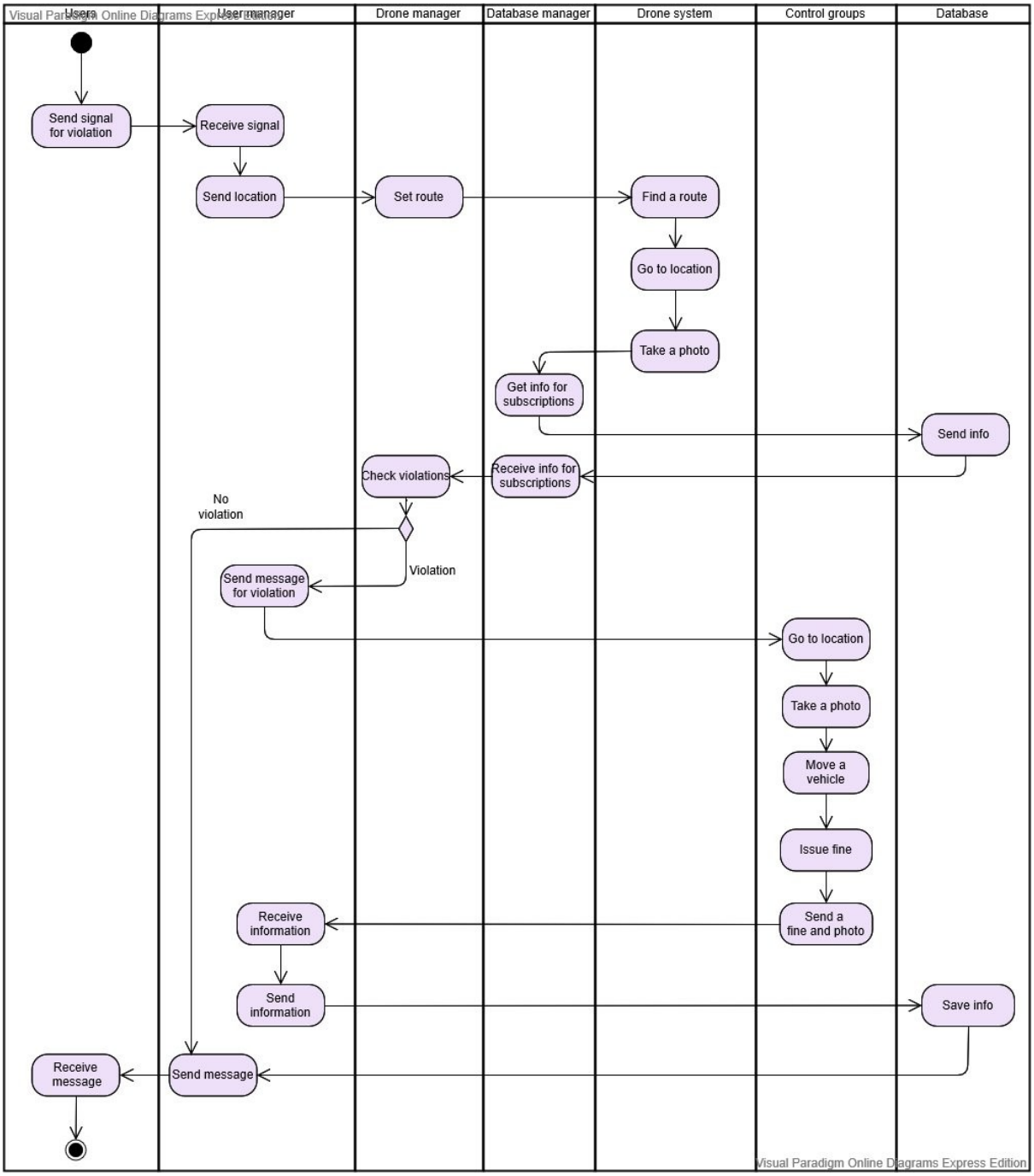
Възможно е добавянето на нов метод за плащане, както и свързването на нашата система и по-точно на модула сървър с базата на МВР с цел автентикация на данните въведени от потребителите, като регистрационен номер на МПС и собственик на МПС.

3. Описание на допълнителните структури

3.1. Структура на процесите

3.1.1. Диаграма, показваща процесите, които се извършват при подаден сигнал за нарушение

3.1.1.1. Първично представяне



3.1.1.2. Описание на елементите и връзките

При подаване на сигнал за нарушение от потребител, User manager в сървъра обработва заявката и изпраща информация на Drone manager за местоположението на нарушителя. Drone manager задава маршрута на Drone system. След като дроновете стигнат до мястото изпращат снимка до Database manager, който трябва да провери дали има нарушение и да върне отговор на Drone manager. Ако такова не бъде установено от Drone manager процесът приключва. В противен случай се вземат мерки

за известяване на групите по контрол от User manager, които освен принудителното преместване на автомобила, заснемат настъпилото събитие и издават електронен фиш за глоба. Снимката и фишът се изпращат до Database manager, а той до базата данни, за да бъдат запазени в архив. Връща се съобщение на потребителя, че сигналът му е приет успешно.

3.1.1.3. Описание на обкръжението

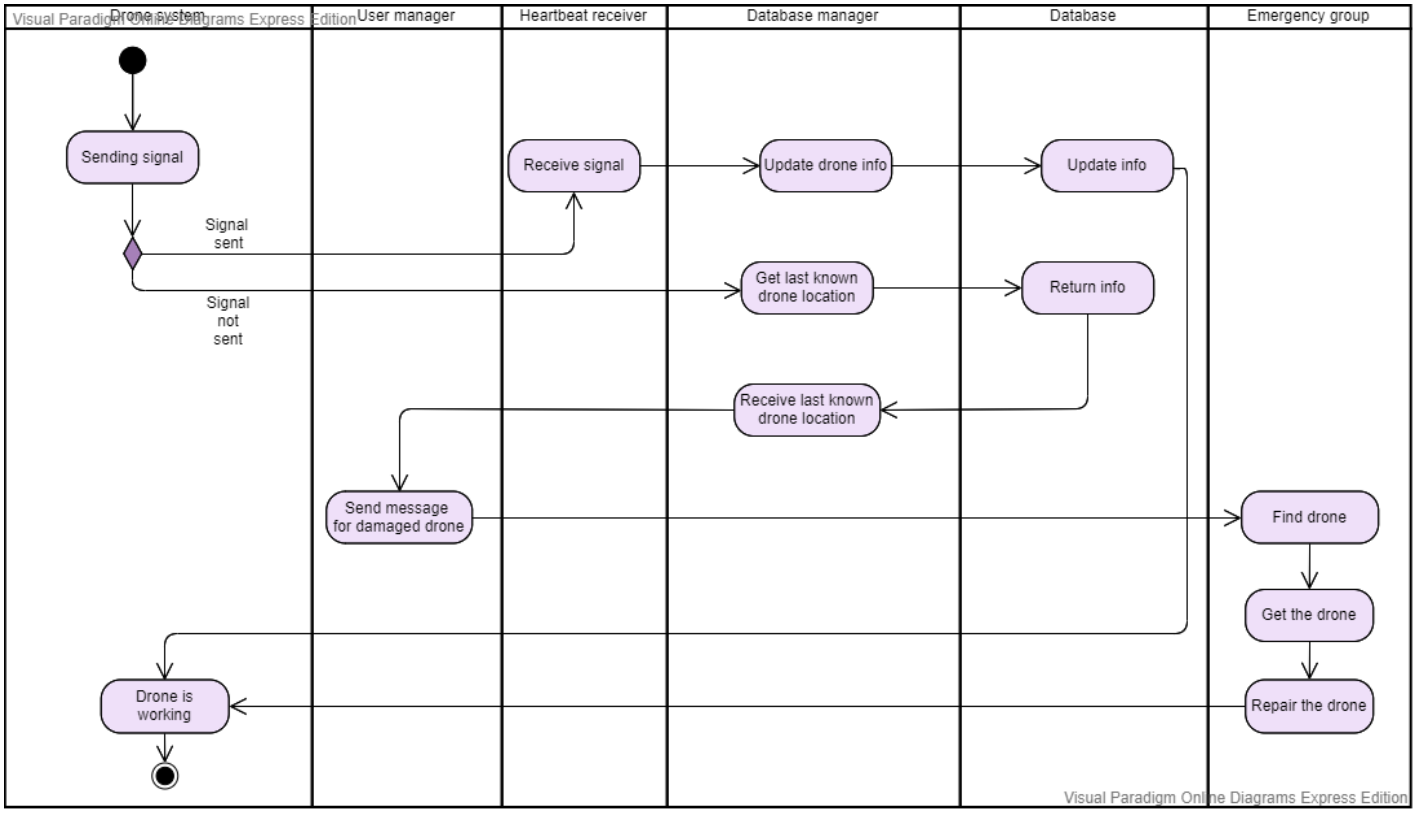
При първоначална проверка за установяване на нарушители на системата използваме външна система Drone System.

3.1.1.4. Описание на възможните вариации

Автоматизация на процеса на издаване на фишове.

3.1.2. Диаграма, показваща процесите, които се извършват при повреда на дрон

3.1.2.1. Първично представяне



3.1.2.2. Описание на елементите и връзките

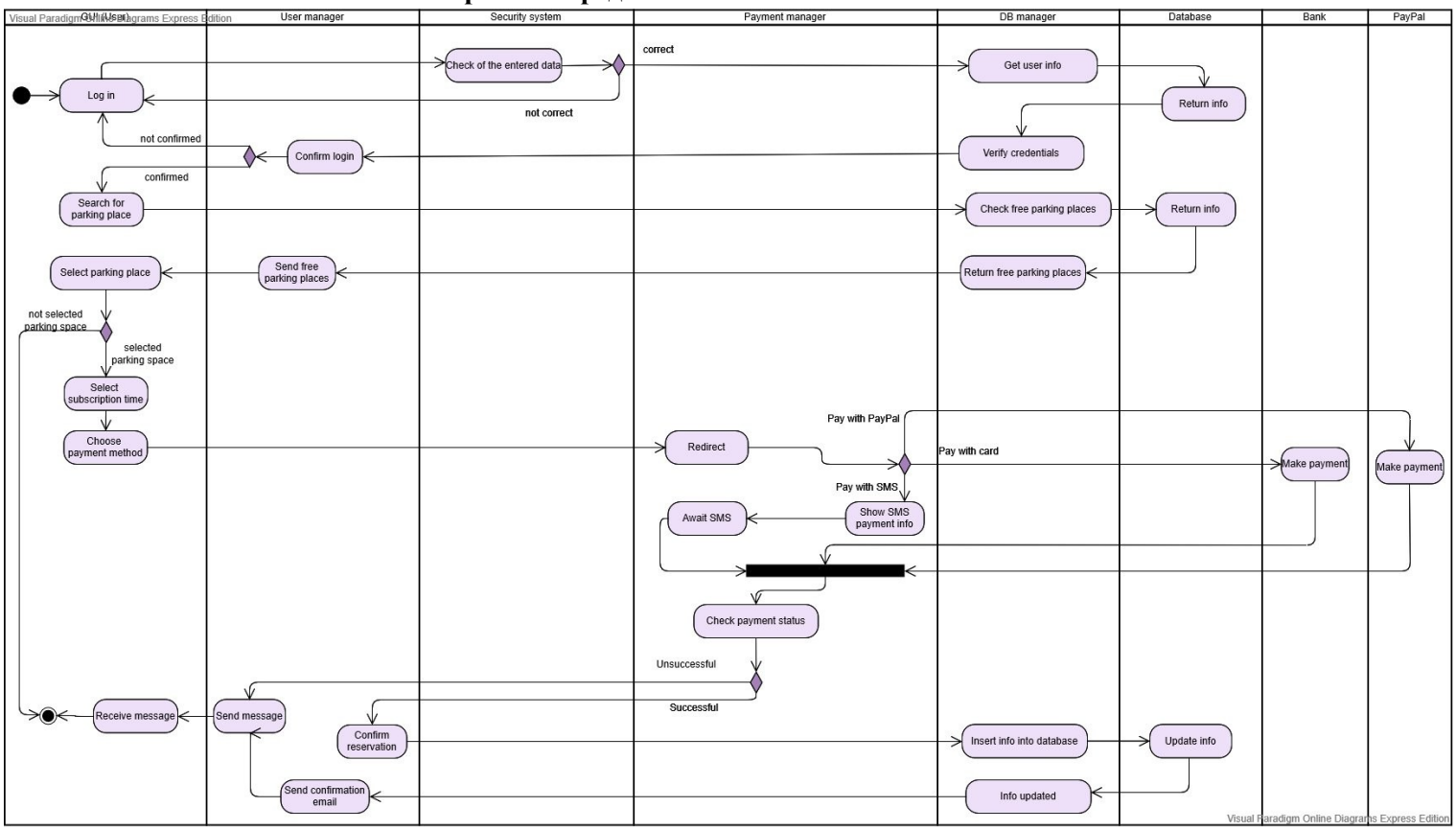
С цел предотвратяване на откази в нашата система използваме тактиката Heartbeat, при която през даден интервал от време всеки дрон изпраща сигнал до сървъра и по-точно до Heartbeat receiver, съдържащ информация за местоположението, на което се намира. Подмодулът обработва сигнала и се свърза с Database manager, които изпраща в базата данни информация за местоположението на дрона. Ако обаче такъв сигнал не бъде изпратен от някои дрон, това означава, че е настъпила повреда. Database manager прави заявка към базата данни относно последната известна локация на дрона, след което уведомява групите по контрол и им изпраща нужната информация. Те намират дрона, взимат го и отстраняват повредата.

3.1.2.3. Описание на обкръжението

В този процес нашата система си взаимодейства с Database и Drones System. Дроновете изпращат сигнал, който сървърът обработва.

3.1.3. Диаграма, показваща процесите, които се извършват, когато потребител иска да се абонира за паркомясто

3.1.3.1. Първично представяне



3.1.3.2. Описание на елементите и връзките

При вход на потребител Security system проверява въведените от него символи дали са валидни. Ако не са системата връща потребителя за нов опит. В противен случай DB manager прави заявка към базата данни за верификация на данните. При несъответствие на данните, User manager отново връща потребителя на вход, иначе се предоставя функционалност, даваща възможност за търсене на свободно паркомясто. DB manager прави заявка за наличните в момента паркоместа към базата данни и се извежда резултатът. Ако потребителят не си избере паркомясто, процесът приключва. Иначе той трябва да избере периода на абонамента и метод за плащане. Payment manager се свързва с външни системи за плащане, като ако избере SMS метода, потребителят има определен интервал от време, в който може да изпрати съобщение с предоставени от системата съдържание и номер за връзка. Изчаква се потвърждение на плащането, като ако то бъде неуспешно, процесът на резервация на паркомясто приключва. При успешно плащане, потребителят успява да запази своето паркомясто, а DB manager записва промените в базата данни. User manager изпраща съобщение на потребителя дали резервацията му е била успешна или не.

3.1.3.3. Описание на обкръжението

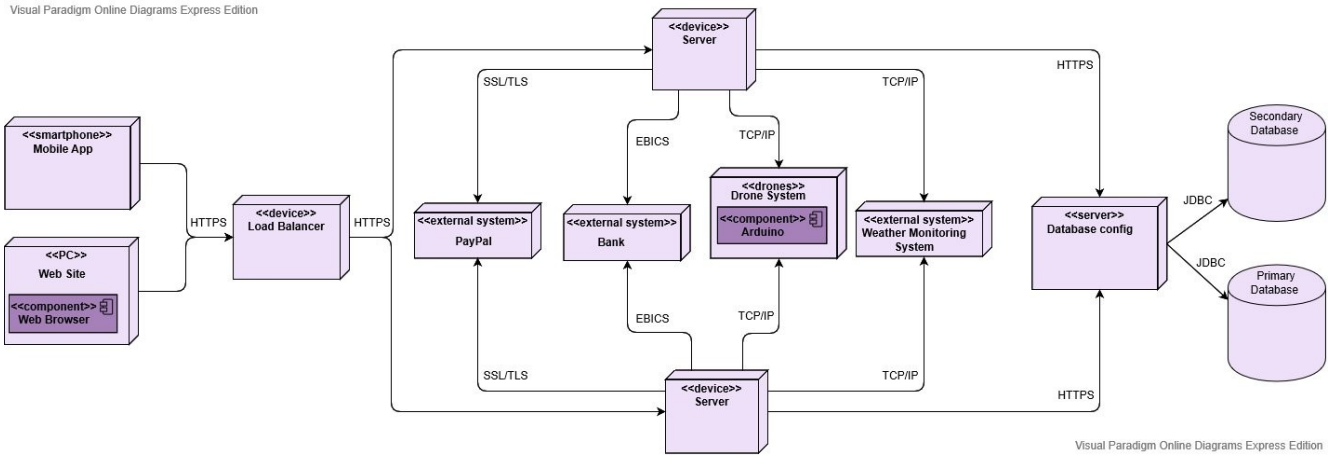
При плащане в системата се обръщаме към външните системи PayPal и Bank, от който чакаме потвърждение относно плащането.

3.1.3.4. Описание на възможните вариации

Възможно е да се добавят и други начини на плащане.

3.2. Структура на внедряването

3.2.1. Първично представяне



3.2.2. Описание на елементите и връзките

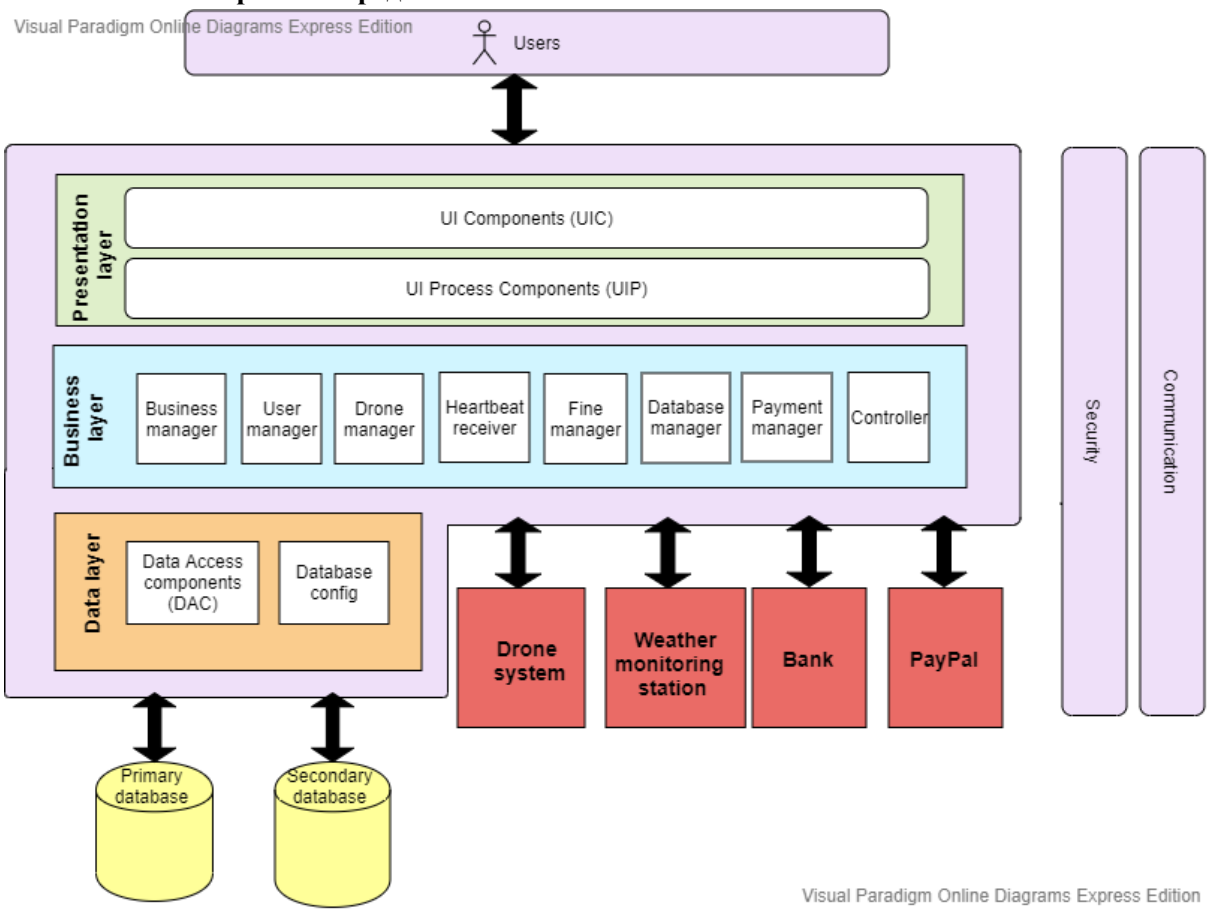
Нашата система има потребители, сървър и база данни. Потребителят използва системата с помощта на уеб браузър или с мобилно приложение, които комуникират чрез защитения протокол HTTPS с Load Balancer, който разпределя заявките от потребителите между двете инстанции на сървъра. Това се случва с помощта на HTTPS протокол. С цел отказоустойчивост на нашата система и двата сървъра са свързани поотделно с външните системи чрез съответните протоколи - TCP/IP (Drone System,Weather Monitoring System), EBICS (Bank), SSL/TLS (PayPal), както и с HTTPS Database config. Database config комуникира с двете инстанции (използваме тактиката активен излишък) на базата данни чрез JDBC.

3.2.3. Описание на възможните вариации

Възможна е интеграцията на друга външна система за плащане, както и друга система за верификацията на потребителските данни.

3.3. Многослойна структура

3.3.1. Първично представяне



3.3.2. Описание на елементите и връзките

Най-горния слой на приложението е UI. Основната роля на интерфейса е да показва резултати от извършените процеси в системата по начин, по който потребителят разбира.

Бизнес слоят координира приложението, изпълнява команди, прави логически решения и оценки, извършва изчисления, свързва се с външни системи. Също така мести и обработва данни между UI и базата данни.

Data layer слоя служи за съхранение на информация за конфигурационните файлове и извличане/записване на информация от/в базата данни.

3.3.3. Описание на обкръжението

Нашата система се свързва със система от дронове, която идентифицира свободните паркоместа и заснема зоните за паркиране. На базата на информация взета от Weather monitoring system, сървърът определя маршрута на дроновете. С Bank и PayPal се свързва при плащане от потребителя по съответния начин.

3.3.4. Описание на възможните вариации

Възможно е добавянето на друга система за плащане.

4. Архитектурна обосновка

*Забележка: Подчертаните архитектурни изисквания са избраните от нас драйвери. В първата част се представят архитектурните решения и компонентите, които са включени в нашата система, а втората част съдържа аргументация за избора ни за архитектурни драйвери.

4.1. <u>Свободните паркоместа се идентифицират от система от дронове, които обикалят града и заснемат зоните за паркиране (отгоре).</u>

За определяне на свободните паркоместа интегрираме външна система от дронове, която използва камера, чрез която заснема зоните за паркиране. Данните от заснетите изображения се изпращат до Drone manager в сървъра, който определя свободните паркоместа и се свързва с Database manager, който записва резултата в базата данни в таблица Parking spaces.

Това изискване е ключово за архитектурата, понеже в него се състои основното ѝ предназначение, както предоставя информация за използването на система от дронове. Именно системата от дронове осигурява основната функционалност. Те трябва да работят точно, ефикасно и безотказно, за да не предизвикат неточности и проблеми, които може да се окажат фатални.

4.2. Броят на летящите в момента дронове и маршрутът на всеки от тях се определя динамично, на базата на предвиждане, за честотата на заемане/освобождаване на места в съответните зони. Това предвиждане зависи от натрупаните данни за динамиката на паркиране в съответния ден и час от седмицата и метеорологичните условия.

Нашият Drone manager определя броя на летящите в момента дронове и маршрута на всеки от тях, използвайки данни от метеорологичната система и данни от предходни дни за динамиката на паркирането, които се записват в базата данни в Parking Dynamics. На базата на тази информация ежедневно се прави статистика, която се пази отново в Parking Dynamics.

4.3. За определяне на метеорологичните условия да се ползва външна услуга за прогноза за времето.

Данни за прогнозата за времето се изпращат до сървъра от Weather monitoring system, на базата на които Drone manager определя маршрута на дроновете, а Business manager цената на паркоместата.

4.4. Системата използва специфичен алгоритъм за разпознаване на свободните места, на база на заснетите изображения.

Drone Manager чрез специфичен алгоритъм разпознава свободните паркоместа на база снимките, изпратени от системата от дронове. След което Database manager записва данните от изчисленията в Parking spaces в базата данни.

4.5. <u>Системата поддържа следните групи потребители:</u>

- **<u>Администратор</u>**
- **<u>Оператор</u>**
- **<u>Аварийни групи</u>**
- **<u>Групи по контрол на паркирането (т.нар. „паяци“)</u>**
- **<u>Регистрирани потребители</u>**
- **<u>Обикновени потребители</u>**

Нашата система има Web Site и Mobile App, които имат UI за съответните групи потребители, след като те са влезли в системата. Информация за потребителските акаунти пазим в User Accounts. User manager се грижи за регистрацията и логването на потребителите. Също така при вход на потребител изпраща заявка до Authentication мениджъра, за да бъде идентифициран потребителят и Authorization manager, който проверява дали потребителят има достъп до определени ресурси.

Архитектурата предоставя общ език за всички заинтересовани лица. За изграждането ѝ трябва да се знае кои ще бъдат потребителите на системата. Разделянето им на групи трябва да се направи още в началния етап от изграждането на системата. Това се прави с цел уточняване на правата на всяка група, за да може системата да се разработи спрямо тях, и да се определят основните им функции. Поради тази причина това изискване е драйвер.

4.6. <u>Ако някой дрон излезе от строя, незабавно трябва да се уведомят аварийните групи, които да получат информация за предполагаемия район, в който се намира дрона и да отстранят повредата.</u>

Описали сме това изискване в точка 3.1.1 "Диаграма, показваща процесите, които се извършват при подаден сигнал за нарушение". За осигуряване на изправност на системата от дронове използваме тактиката Heartbeat. Подмодулът Heartbeat receiver очаква сигнал от дроновете, като ако такъв не бъде получен се известява операторът на системата.

Всяка група потребители трябва да има ясно дефинирани функционалности, на база на които се определят техните права. Това изискване е драйвер, защото предоставя информация за работата на аварийните групи, както за комуникация между дроновете и представителите на аварийните групи.

4.7. Информацията за свободните места се обновява на определен интервал от време, който се задава от оператора на системата и може да е най-малко 1 минута.

На оператора на системата се предоставя функционалност в GUI за задаване на интервал от време. Controller следи за направени заявки от UI и след изтичането на времето, зададено от оператора, се свързва с Database manager за обновяване на съдържанието на Parking spaces в базата данни.

4.8. При трайно намалена видимост (напр. мъгла), която води до невъзможност да се заснемат паркоместата, да се вземат мерки за известяване на оператора на системата.

Drone manager получава информация за прогнозата на времето. При установяване на трайно намалена видимост, той взима мерки за известяването на оператора на системата.

4.9. <u>Регистрираните потребители могат да заплащат абонамент за определено паркомясто, което се маркира като заето в рамките на периода на абонамента, независимо дали заснетите от дроновете изображения, показват наличието на автомобил на него или не.</u>

Този процес е описан в точка 3.1.3 "Диаграма, показваща процесите, които се извършват, когато потребител иска да се абонира за паркомясто". Регистрираните потребители имат функционалност Reservation for parking place в GUI, чрез която се абонират за определено паркомясто, както и функционалност Pay (за плащане на абонамент за свободно паркомясто). В базата данни пазим информация за Parking spaces - информация за всички места за паркиране (дали дадено място е свободно или има платен абонамент). В User subscriptions пазим данни за активните абонаменти.

Тук се представя възможността на всеки регистриран потребител да плати за абонамент. Основната функционалност на нашата система е намирането на свободно паркомясто и предоставянето му на регистрирания потребител. Важна част от архитектурата е извършването на обмен на информация между тези модули. Това изискване дефинира опции за наемане на паркомясто, както и период за използването му.

4.10. Ако няма абонамент, свободните места за паркиране може да са безплатни или да се таксуват динамично, като цената се определя според предвиждане за честотата на заемане/освобождаване в съответния ден/час, както и от прогнозата за времето.

Business Manager определя цената за паркоместата, за които няма абонамент, в определен ден и час на база предходни предвиждания за честотата на заемане на съответното паркомясто, както и от предоставени данни за времето от метеорологичната система.

4.11. <u>Плащането може да се извършва чрез дебитна/кредитна карта, PayPal или СМС, като в бъдеще може да се добавят и други начини на разплащане.</u>

При заявка от потребител за плащане, Payment Manager следи за това, свързва се с външни системи (PayPal, Bank) или предоставя информация за плащане чрез SMS и връща резултат дали е било успешно плащането. Ако резултатът е бил положителен се записва в User Subscriptions.

При разработката на архитектурата трябва да бъде предвиден начин за плащане от потребителите на системата. За тази цел трябва да бъде включена услуга, която да се интегрира, а горе са посочени външни системи, към които ще се обръщаме за изпълнение. Това изискване е важно, защото дефинира начините на плащане от потребителите, за да използват услугите на системата, което е и целта, за която се проектира тя.

4.12. <u>Обикновените потребители, регистрираните потребители, аварийните и групите по контрол на паркирането използват системата през мобилно приложение, като може да заемат само свободните места, за които няма абонамент.</u>

Предоставя се възможност на посочените по-горе групи потребители да използват системата чрез мобилно приложение. При заемане на паркомясто от потребител, трябва да се направи заявка към системата, чрез предоставената функционалност Reservation for parking places за отбелязването на съответното паркомясто като заето в Parking spaces. Регистрираните потребители заплащат абонамент при резервация на паркомясто.

Това изискване е архитектурен драйвер, защото предвижда използването на системата да става и чрез мобилно приложение, с цел по-лесен достъп на групите потребители, за които е предназначено то. За да се изпълни това изискване е необходимо да се разработи мобилно приложение и да се осигури комуникацията между приложението и сървъра.

4.13. <u>Останалите потребители трябва да имат 100% защитен от външна намеса достъп до системата.</u>

Освен използваните тактики при вход в системата - Authentication and Authorization в Security system имаме модул Admin and operator security. С цел 100% защита на потребителите от тези групи, техният достъп до системата ще е възможен само от определени устройства и IP адреси.

По принцип това изискване трябва да бъде драйвер на всяка една система, но за нашата е от изключително важно значение, тъй при неправомерен достъп до функционалностите на системата (в частност управлението на системата от дронове) може да има сериозни последствия. Освен посочените основополагащи функционалности, които системата трябва да предоставя, се налага и съобразяването с някои изисквания за защита. Нужно е предприемането на строги мерки за правилно съхранение.

4.14. <u>Групите по контрол на паркирането следят дали няма нарушители (неплатили или заели място, за което нямат абонамент). При засичане на нарушител, освен принудителното преместване на автомобила, заснемат настъпилото събитие, като снимката се съхранява директно в системата и след това се издава електронен фиш за глоба. Снимката и фишът трябва да са достъпни и през публичен сайт, който се зарежда чрез уеб-браузър.</u>

Този процес е представен в точка 3.1.1 “Диаграма, показваща процесите, които се извършват при подаден сигнал за нарушение”. При издаване на фиш, Fine manager записва информацията в Violations. В Violations пазим снимка на нарушението, електронен фиш. Тази информация е достъпна през публичен уебсайт.

Ключово за системата изискване е следенето за нарушители, като от горното изискване става ясно коя група отговаря за това и се уреждат мерките, които трябва да бъдат взети. Също така се определя какво правят отделните модули от архитектурата при наличието на нарушител, както и комуникацията помежду им. От това изискване се разбира и за наличието на сайт, т.е системата да бъде достъпна не само от мобилно приложение, а и от уеб браузър. Така се осигурява по-широк достъп до нашата система.

4.15. Регистрираните потребители може да подават сигнал до групите по контрол на паркирането за неправомерно заето от друг място, за което са абонирани.

Този процес е представен в точка 3.1.1 “Диаграма, показваща процесите, които се извършват при подаден сигнал за нарушение”.

4.16. <u>Системата да работи 100% без отказ в рамките на светлата част на работния ден (9:00 до 17:00 зимно време и 8:00 – 19:00 лятно време).</u>

С цел системата да работи 100% без отказ се дублира базата данни (активен излишък). Също така имаме Load balancer, който разпределя заявките от потребителите върху двата сървъра с цел по-висока производителност. Също така използваме тактиката Heartbeat, за предотвратяване на откази от страна на системата от дронове.

Необходимо е системата да работи без отказ, като в архитектурата трябва да се включат различни компоненти с цел предотвратяване на прекъсвания, които биха имали нежелани последствия. Това изискване е драйвер, защото предоставя информация, на база на която трябва да се предвидят архитектурни решения за осигуряването на това качествено изискване.

4.17. <u>Системата да поддържа архив на данните за динамиката на паркирането и всички издадени фишове за глоби за 25 години назад във времето, както и архив на заснетите изображения за 3 години назад.</u>

В базата данни имаме Archive, където се пази тази информация.

Системата трябва да включва база от данни, в която да се пазят фишове, както и заснетите изображения. Периодът на съхранение на данните е дълъг, което прави това изискване архитектурен драйвер. Предвижда се системата да изтрива автоматично старите данни. Поддържането на архив с доказателства за нарушенията е важно, в случай на обжалване от страна на потребител. Също така системата може да идентифицира честите нарушители, като се прави справка на издадените фишове 25 години назад.