**Модул за справки към Детелина през мобилен телефон**

**Задание за разработка**

1. **Общи положения**
   * Системата визуализира справки от произволна база данни на Детелина през мобилен телефон
   * Поддържат се следните операционни системи (за телефон): Android; IOS; Wndows Phone. От страна на Детелина: Windows 7;8;10
   * За връзка между базата данни на Детелина и мобилното устройство се използва налична интернет връзка. Не може да се рзчита на статичен и реален IP адрес. Обикновено базата данни на Детелина се намира зад firewall.
   * За да се изпълни горното условие се използва мождинен комуникационен сървър (облак). На този сървър не се пазят клиентски данни. Там се пази единствено инормация за самия билинг.
   * Системата е проектирана да работи „под наем“. Тоест поддържат се обонаменти за период от време. Ценообразуването не е част от тази система. Билинга е на ниво сметка и дата до която може е активна. Под сметка се разбира отделна база данни на Детелина. За всяка база данни на Детелина се заплаща отделен абонамент. Тоест ако клиент иска да наблюдава някоко бази данни на Детелина, трябва да заплати абонамент за всяке една от тях. (таксуването е на нивп база данни, а не на потребител)
   * Абонамента може да се подновява както през самата система така и през външна система посреством API.
2. **Топология на системата** (схема и устройства)
   * **Клиентски компютър** – Windows базирано PC на което се хоства базата данни на самата Детелина. Базата може да бъде на Детелина лайт, или Детелина. Също така базата данни може да бъде както централна така и починена (на обект). За връзка се използва наличен интернет без специални изисквания.
   * **Мобилен телефон** – Използва се за визуализиране на самите справки. Може да показва справки от различни бази данни на Детелина. От меню, потребителя избира към коя база да се свърже. Абонамента се следи на ниво база данни.
   * **Облак на Елтрейд** – Windows базиран сървър със статично и реално IP. За връзка към този сървър се използва DNS име (например cloud1.eltrade.com). Служи за осъществяване на връзка между горните две устройства. Освен това този сървър се грижи за самия билинг (плащания и абонаменти). На този сървър не се съхраняват клиентски данни. Единствените данни които пази сървъра са тези за билинга.
3. **Принцип на работа**
   * Справките са WEB базирани. Не се ползва стандартния браузер на телефона, а вместо това нейтив приложение покзва самите веб страници (това се прави за да идентифицираме еднозначно устройството)
   * WEB сайта чете данните от отдалечените бази посредством TCP сървър и клиенти. (все едно че прави заявка към локална база) Когато WEB сайта иска някакви данни, подава SQL заявка и ID на базата данни. TCP сървъра се грижи да намери съответния клиент и да изиска данните от него. WEB сайта и TCP сървъра комуникират помежду си с HTTP заявки (XML съдържание).
   * Всяка една база данни на Детелина трябва да е уникална за системата (уникално ID). Освен това има и потребителска парола (понеже IDто е видимо)
   * Всяко едно мобилно устройство трябва да бъде уникално пред системата. По този начин всички настроки се пазят в сайта а не в телефона. (самия сайт ще знае какво да покаже на потребителя; кои бази си е добавил за гледане)
   * Не използваме потребителско име и парола за идентификация на телефона. Вместо това трябва да идентифицираме еднозначно самото устройство посредством уникален номер + служебна парола
   * След като сайта идентифицира мобилното устройство, показва на потребителя всички въведени бази данни на Детелина, които могат да бъдат достъпвани
   * Целта на идентификацията на устройството е да не се пазят никакви данни в самото него. Всички настройки и конфигурации са на сървъра. По този начин локалното (нейтив) приложение става унивеерсално и няма да има нужда от ъпдейти за нови функции (къстомизации на справки).
4. **Сигурност и достъп до данните**
   * Мобилното устройство достъпва системата като се идентифицира с уникалния си номер. За да се избегне подслушване, номера не се праща в явен вид. Използва се криптиране (каквото може да се постигне на самото устройство). Криптираме фиксиран стринг + номера + системното време. По този начин заявката всеки път ще има различен идентификационен ключ. На базата на този номер сървъра идентифицира устройството и показва персонализирани менюта.

Трябва да се измисли вариант при който не може да се повтори една и съща заявка.

Тоест ако някой подлуша комуникацията и дублира заявката, не трябва да получи достъп до системата

Пример:

* мобилното устройство иницира сесия. Посредством REST API с обръща към сървъра и иска ключ с който да криптира информацията. (Ключа може да е позиция и дължина на текст от предварително зададен файл който е еднакъв на двете машини)
* сървъра връща две числа: позиция и брой байтове на ключа с който криптираме. (В краен случай може ключа за криптиране да се връща от директно от сървъра – под хормата на дата и час или нещо пдобно
* С получения ключ от сървъра се криптира номера на устройството + фиксиран стринг
* Криптирания резултат се ползва за оторизация пред сървъра ....

ГОРЕНАПИСАНОТО НЕ Е ОКОНЧАТЕЛНО. МОЖЕ ДА ИЗМИСЛИМ И ПО ДОБЪР ВАРИАНТ

ВАЖНО Е ДА ЗАЩИТИМ ДАННИТЕ ОТ ДОСТЪП ЧРЕЗ ПОДСЛУШВАНЕ И ИЗПОЛЗВАНЕ НА ОГЛЕДАЛНИ ЗАЯВКИ

* + Добавянето на база данни се осъществява като се въведе ID на базата и парола. Веднъж добавена база към устойство, се запаметява и се показва във спсиъка с обекти
  + Комуникацията между мобилното устройство и сървъра не е криптирана. Ако успеем да намерим сертификат на поносима цена може да исползваме HTTPS
  + Комуникацията между сървиса на детелина и облака е по TCP като се пренасят криптирани и компресирани XMLи. Сървиса отговаря на заявките при правилно подадени ID на база данни и парола. (IDто и паролата се пазат в самата база данни)
  + Комуникацията мижду TCP сървъра и HTTP сървъта намиращи се на облака не е криптирана. Все пак става въпрос за локална комуникация между приложения на една машина

1. **Набор от справки и къстомизация на системата**
   * Оборот за деня + графика за последните 5 дни
2. **Интерфейс на системата**

Веб сайт позволяващ администрация на системата и набор от справки

* + Достъпа до системата е посредством потребител и парола
  + Интерфейс за подновяване на абонамент . За търсене се използва име на фирмата или ID на базата данни.
  + Справка регистрирани бази данни; статус (онлайн/офлайн) ; дата и час на последен достъп...
  + Статистика на достъпа до сървъра от страна на мобилни устройства и бази данни
  + REST апи за интеграция с външни системи. Използва се за подновяване на абонамент

1. **Модули на системата**
   * **Сървиз за връзка с базата данни на Детелина**

Win32 Service Application. Средство за разработка DELPHI 7. Осъществява връзка с облака посредством TCP протокол. При пускането си се закача на облака и очаква заявки за изпълнение към базата данни. Връзката с облака се поддържа непрестанно. Използва се TCP протокол за комуникация. Предават се XML криптирани и компресирани съобщения. Интерфейса е универсален. Облака подава заявка, която се изпълнява от сървиса като отговора се връща обратно. Самата заявка съдържа една или повече SQL заявки. Оторизацията (идентификацията) на базата данни се осъществява посредством уникално ID генерирано от облака. Самото ID се пази в базата дани (това позволява запазването му при копиране на базата).

Алгоритъм за генериране ID нa база: При пускане на сървиса с нова база (в която няма записано ID) , сървиса заявява генериране на ново ID от облака посредством REST заявка. След като получи ID сървиса го запазва в базата данни и го ползва при свързване. От тук натам въпроснот ID се ползва за идентификация на базата данни пред облака. При пускане на сървиса със стара база (вече има ID) се прави проверка дали това ID e валидно. Освен това се изпраща информация за самата база данни (име на обект, фирма и т.н.). Тази информация се ползва за показване в справките на самия телефон. При валидацията се следи дали няма дублирани IDта на бази данни. Сървис с потарящо се ID не се допуска на облака (това може да се случи при копиране на базата данни). При същата проверка се следи дали има валиден абонамент.

В резюме: сървиса комуникира с TCP комуникациония сървър и HTTP комуникациония сървър на облака. ПО TCP се предават самите заявки. По HTTP се предава служебната информация. На практика достъп до базата данни на облака има само HTTP сървъра.

За ускорение на бързодействието и стабилността TCP сървъра не ползва база данни и не прави валидации.

Освен ID e добре да имаме и парола, която се задава от потребителя. Паролата не се вижда от всички за разлика от IDто. Без правилна парола сървиса не праща данни.

* + **Административен интерфейс на DB сървиса**

Реализира се като operation dll към Детелина. Показва статус на комуникационния сървис. Позволява промяна на ID то на базата с нововъведено от оператора (в случай че го знае, или е платено). Освен това позволява и ресет на ID така че сървъра да даде ново. (в случай че сме копирали база със записано ID и искаме да го ресетнем). От тук се сменя и паролата за връзка с базата данни. По подразбиране паролата има стойност „12345“

* + **Статус приложение на DB сървиса**

Стандартен Windows application. Крие се при часовника и показва дали сървиса е активен в момента (дали има връзка с облака). Освен това може да показва дата и час на кативиране, брой изпълнени заявки към базата. Добре е да показва и текущото ID на базата за да знаем към коя база да се вържем през телефона. Паролата не се показва.

* + **Инсталатор на сървиса и админ интерфейса за Детелина**

Модула трябва да може да се инсталира към всяка една Детелина. Тоест трябва да бъде включен в стандартния сетъп на бек офиса с възможност за избор. Освен това трябва да има и отделна инсталираща програма (за съществуващи Детелини). Сетъпа инсталира: сървиса, DLLа и статус приложението

* + **База данни – облак**

MySQL база данни. Пази информация за:

* + - * Мобилни устройства
      * Настройки на мобилните устройства (списък с добавени бази данни)
      * Клиентски бази данни. ID; име на фирма; име на обект; име на PC; …
      * Платежна информация (подновяване на лиценз; дата период)
      * Комуникационни събития
  + **TCP комуникационен сървър – облак**

Delphi базиран Win32 service application. Има два интерфейса. През единия комуникира по TCP с DB сървисите на отдалечените машини. По другия комуникира с HTTP сървъра на самия облак. TCP комуникацията с DB сървисите е архивирана и криптирана. Принципа на работа е следния: При нужда от данни HTTP сървъра ги изисква посредством HTTP заявка. Самата заявка съдържа ID на база данни; парола; SQL заявки (като текст). По самото ID сървиса претърсва списъка с активни конекции. Ако намери правилната изпраща заявката към отдалечената машина за изпълнение. Проверката на паролата се извършва на самата отдалечена машина. При ненамиране на сървис с конкретно ID се връща съобщение за грешка.

TCP сървъра поддържа „живи“ конекции с всички отдалечени DB сървиси. Допускат се само уникални IDта. При дублиране на ID се изхвърлят и двете конекции (понеже не знаем коя е оригиналната) През HTTP интерфейса сървъра връща и информация за текущия статус на закачените клиенти. Използва се за справки. HTTP Интерфейса е локален и не се експортва навън от облака

* + **HTTP комуникационен сървър – облак (web site)**

Apache (php) веб сайт. Зардаи лицензи се спираме на безплатни средства

* + **HTTP RestApi – облак**

Apache (php) . Въпросното API е видимо навън от облака(за разлика от TCP сървиса). Всяка една заявка към апито се логва в базата данни за да има проследимост.

Поддържа следните интерфейси:

* + - * Генериране на ново ID на база данни (използва се от DB сървиса)
      * Валидация на ID и ъпдейт на клиентската информация (фирма, обект, база ...) (използва се от DB сървиса)
      * Проверка на абонамент (по ID на база) (използва се електронен магазин)
      * Подновяване на абонамент (по ID на база) (използва се електронен магазин)
  + **Приложение за телефон – Android**

Визуализира WEB сайта на облака, като реализира подходяща HTTP заявка. За оторизация пред системата се ползва уникален номер на устройството който се криптира

* + **Приложение за стандартен Windows XP;7;8;10;**

Заради особенната оторизация няма как да ползваме стандартния браузер

* + **Приложение за телефон – IOS**

Като при Андроида

* + **Приложение за телефон – Windows Phone**

Като при Андроида

**ПРИЛОЖЕНИЕ 1**

TCP комуникационен сървър – облак

Протокол за комуникация по Http канал

Http сървъра отговаря на POST/GET заявки. Заявката съдържа задача в JSON формат. Задачите се подават като Http content. (На практика сървъра не прави разлика дали заявката е GET или POST)

--> POST /ENDPOINT HTTP/1.1

Host: ...

Content-Type: application/json-rpc

Content-Length: ...

{"method": "sum", "params": {"a":3, "b":4}, "id":0}

<-- HTTP/1.1 200 OK

...

Content-Type: application/json-rpc

{"result": 7, "error": null, "id": 0}

1. **Заявка за получаване на данни от сървиса на Детелина**

Самата задача (JSON) се изпраща директно към сървиса на Детелина без да се валидира на ниво сървър. Сървиса на Детелина подготвя данните и ги връща обратно в Json формат.

ENDPOINT:

<http://127.0.0.1:8001/report/nameofreport/?id=1&u=httpuser&p=httppass>

PARAMETERS:

**id** – уникален идентификатор на Детелина

**u** – потребителско име за достъп до HTTP сървъра

**p** – парола за достъп до HTTP сървъра

Самата задача която ще се изпрати до клиентския сървиз на Детелина трябва да се подаде като контент на get заявката. Задачата е във JSON формат от следния вид:

{

"Id":"DatabaseId",

"Pass":"DatabasePass",

"Q1":

{

"Type":"Query",

"SQL":"select....."

},

"Q2":

{

"Type":"Query",

"SQL":"select ...."

}

"Cmd1":

{

"Type":"Command",

"Param":"ShutDown"

}

}

Отговора на задачата е от вида:

{

"ResultCode":0,

"ResultMessage":"OK",

"Q1":[

{"OPT\_ID":17,"OPT\_NAME":"EnScales","OPT\_STRING":""},

{"OPT\_ID":1,"OPT\_NAME":"TaxGrp","OPT\_STRING":"2"},

{"OPT\_ID":2,"OPT\_NAME":"PrnGrp","OPT\_STRING":"1"},

{"OPT\_ID":3,"OPT\_NAME":"MeasureID","OPT\_STRING":"1"}

]

"Q2":[

{"ID":17,"VALUE":"EnScales"},

{"ID":18,"VALUE":"EnScanner"}

]

"Cmd1":{

"ResultCode":0,

"ResultMessage":"OK",

}

}

Отговор на задачата при неуспешно изпълнение:

{

"ResultCode": 103,

"ResultMessage": "HTTP authorisation fail! Access denied!",

}

Списък грешки:

C\_HttpErr\_MissingClientID = 100;

C\_HttpErr\_MissingClientPass = 101;

C\_HttpErr\_MissingLoginInfo = 102;

C\_HttpErr\_LoginIncorrect = 103;

C\_HttpErr\_ClientIsOffline = 200;

C\_HttpErr\_ClientIsByssy = 201;

1. Заявки за получаване на данни от TCP сървъра на облака

Самата заявка (JSON) се обработва от сървъра на облака без да се изпраща към клиентския сървис на Детелина. Използва се за извличане на служебна информация от облака.

Възможните заявки са следните:

* + Извличане на списък със всички активни клиенти:

ENDPOINT:

<http://127.0.0.1:8001/server/clientlist/?u=httpuser&p=httppass>

PARAMETERS:

**u** – потребителско име за достъп до HTTP сървъра

**p** – парола за достъп до HTTP сървъра

{

"ResultCode":0,

"ResultMessage":"OK",

"Clients":[

{"Id":"3769d93a",

"Host":"DLUKAREV",

"Conn":"2016-01-03 22:53:16",

"Act":"2016-01-03 22:58:24",

"Name":"Булгартабак Трейдинг АД"

}

]

}

* + Извличане на състоянието за определен клиент (сървиз на Детелина)

ENDPOINT:

<http://127.0.0.1:8001/server/clientstat/?id=1&u=httpuser&p=httppass>

PARAMETERS:

**id** – уникален идентификатор на Детелина

**u** – потребителско име за достъп до HTTP сървъра

**p** – парола за достъп до HTTP сървъра

{

"ResultCode":0,

"ResultMessage":"OK",

"Clients":{

"Id":"3769d93a",

"Host":"DLUKAREV",

"Conn":"2016-01-03 22:53:16",

"Act":"2016-01-03 22:55:10",

"Name":"Булгартабак Трейдинг АД"

}

}

1. сс

**ПРИЛОЖЕНИЕ 2**

Справки:

**Общ оборот**

ENDPOINT:

<http://127.0.0.1:8001/report/dayturnover/?id=3769d93a&u=user&p=pass$123>

при празна заявка:

{

"ResultCode":204,

"ResultMessage":"[TCPC][SendRequest]Data is empty!"

}

при заявка от вида:

{

"Id":"3769d93a",

"Pass":"1234",

"CurrentTurnover":

{

"Type":"Query",

"SQL":"select sum(S.SELL\_SUM) \"SELL\_SUM\", C.CURR\_SIGN

from SALES\_BON S

left join N\_CURRENCY C on S.SELL\_CURRTYPE = C.CURR\_ID

where (cast(S.SELL\_DATETIME as date) = CURRENT\_DATE)and

(S.SELL\_REVOKED\_ = 0) group by C.CURR\_SIGN"

}

"LastTurnover":

{

"Type":"Query",

"SQL":"select cast(S.SELL\_DATETIME as date) \"SELL\_DATE\",

sum(S.SELL\_SUM) \"SELL\_SUM\", C.CURR\_SIGN

from SALES\_BON S

left join N\_CURRENCY C on S.SELL\_CURRTYPE = C.CURR\_ID

where (cast(S.SELL\_DATETIME as date) >=

(CURRENT\_DATE - 7))and

(S.SELL\_REVOKED\_ = 0) group by 1, 3"

}

}

Отговор когато няма данни:

{

"ResultCode":0,

"ResultMessage":"OK",

"CurrentTurnover":[],

"LastTurnover":[]

}

Отговор с данни:

{

"ResultCode":0,

"ResultMessage":"OK",

"CurrentTurnover":[

{"SELL\_SUM":651.87,"CURR\_SIGN":"лв"}

],

"LastTurnover":[

{"SELL\_DATE":"08.02.2013","SELL\_SUM":20.83,"CURR\_SIGN":"лв"},

{"SELL\_DATE":"09.02.2013","SELL\_SUM":244.69,"CURR\_SIGN":"лв"},

{"SELL\_DATE":"11.02.2013","SELL\_SUM":603.34,"CURR\_SIGN":"лв"},

{"SELL\_DATE":"12.02.2013","SELL\_SUM":611.05,"CURR\_SIGN":"лв"},

{"SELL\_DATE":"13.02.2013","SELL\_SUM":650.36,"CURR\_SIGN":"лв"}

]

}