Курсова работа

Софтуерни архитектури и разработка на софтуер

UniDigit

Система за управление на процесите и студентската информация в един университет

Изготвили:
Любка Ангелинина 62342
Николета Вълчинова 62322

1.Въведение
а) обща информация за текущия документ

i. Предназначение на документа

Документът има за цел да представи архитектурата на система за управление на процесите и студентската информация в университет. Показва различните характеристики на системата, спомага за повишаване на качеството на продукта, улеснява работата при поддръжка и изпълнение на изискванията. Така рискът от неспазване на изискванията е редуциран. Документът е предназначен за всички заинтересовани лица по проекта. Структурата му е с цел да улесни разнообразните читатели, когато търсят необходима информация.

ii. Описание на използваните структури на архитектурата.

1. Декомпозиция на модулите
Показва системата, разделена на отделни модули, които са: Клиент, Сървър, База Данни и външни системи, с които си комуникира. Осигурява добра сигурност, обуславя възможност за лесна промяна и ще послужи за разпределянето на работата между екипите.
Клиент е модул, чрез който потребителят взаимодейства със системата.
Сървър е модул, който осигурява бизнес логиката и връзката с базата данни и външните системи.
Базата Данни съхранява всички данни за системата.
Външни системи е модул, който обединява всички външни системи, с които системата трябва да прави връзка. (Система за управление на учебното съдържание, Държавни публични регистри за текущи студенти и т.н.).

2. Структура на процесите
Изобразяват се някои от важните процеси на системата. Структурата на процесите е от особено значение за разглежданата система. По този начин се определя как процесите, реализирани в системата, се свързват помежду си. Това помага в изграждането на системата, за да може да се постигне бързодействие. Чрез структура на процесите, която описва по какъв начин протича последователността на процесите, се показват възможните изходи от даден процес и започването или прекратяването на такъв. Така по-лесно се разбира бизнес-логиката на системата. Структурата показва какви са връзките между компонентите и конекторите. UniDigit има много функционалности, а така заинтересованото лице бързо ще може да се ориентира и да разбере начина на работа на сайта.

3.Структура на внедряването
Показва връзките между софтуерните елементи със средата, в която ще бъде използвана системата. Структурата на внедряване се използва, за да се покаже как софтуерът се разполага върху хардуера и комуникационното оборудване. Също така определя колко сървъра ще са ни нужни и коя функционалност върху кой сървър да бъде внедрена. Тя играе важна роля за изграждането на системата и чрез нея се постига бързодействие и сигурност на системата. Също така и гарантира надеждността ѝ. Изборът ни за именно тази структура

беше воден от факта, че системата ни трябва да може да работи едновременно с много потребители, и следователно е необходима хардуерна поддръжка за обработването на многото заявки.

iii. Структура на документа

1. Въведение - описва се информация за документа: целта и структурата на документа, общи сведения за системата и терминологичния речник.
2. Декомпозиция на модулите - включва първичното представяне на обособените модули, подробно описание на всеки от тях и възможните вариации.
3. Допълнителни структури - тук са разгледани Структурата на процесите и Структурата на внедряването; съдържа първичното представяне, описание на елеметите и връзките в тях, обкръжението им и възможните вариации.
4. Архитектурна обосновка - защитатават се взетите архитектурни решения, въз основа на изискванията и са описани използваните тактики, чрез които са постигнати.

b) Общи сведения за системата

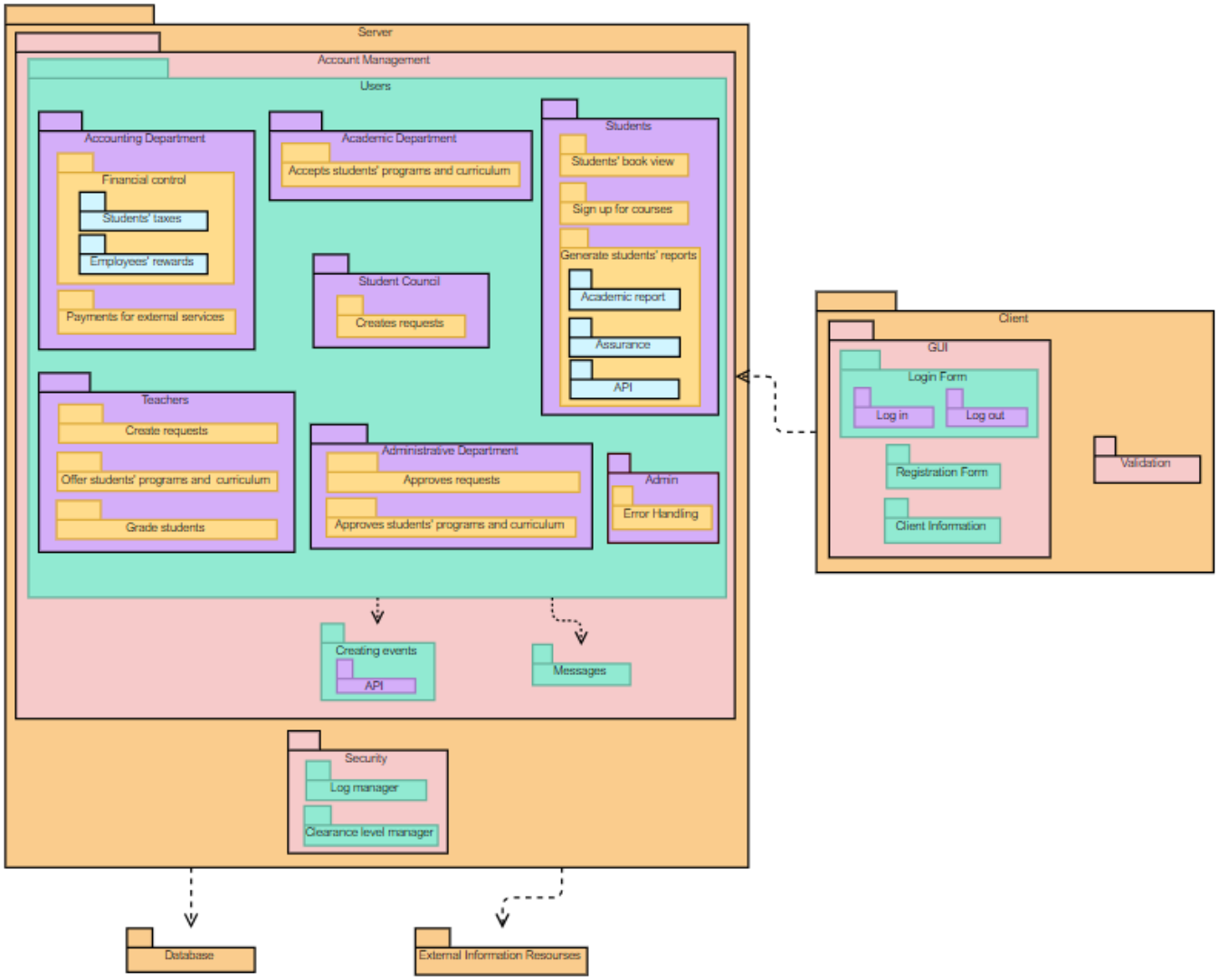
UniDigit е система за управление на процесите и студентската информация в един университет. Системата е предназначена да обслужва преподаватели, студенти както и различните отдели в университета. Системата предлага на клиента различен тип профили, в зависимост от длъжността, която изпълнява.(като не може безразборно да се правят профили - това го правят съответни лица). Различните видове профили имат различни функционалностти. Системата поддържа обмяна на съобщения между всички потребители. Също така UniDigit осигурява защита на личните данни на потребителите.

c) Терминологичен речник

- API (Application Programming Interface) – приложно-програмен интерфейс на изходния код, който се предлага за поддръжката на заявките от приложния софтуер или компютърните програми
- GUI (Graphical User Interface) – графичен потребителски интерфейс, в който елементите, които са продоставени на потребителя за управление, са във вид на графични изображения (икони, които подсказват предназначение и свойства). Спомага за разбиране и усвояване.
- ID (IDentification) – идентификационен номер
- JDBC (Java DataBase Connectivity) - междуплатформен и поддържащ много СУБД програмен интерфейс за използване на бази от данни от програми на Java.
- RMI (Remote Method Invocation) – как обекти на различни компютри могат да си взаимодействат в разпределена мрежа
- UI (UserInterface) – потребителски интерфейс, чрез който потребителят се възползва от фунционалността на програмата. Той дава възможност на потребителя да въведе входни данни и да получи резултати.

2. Декомпозиция на модулите

a) Общ вид на декомпозицията на модули за системата



b) Контекстна диаграма (това е т.нар. описание на обкръжението)

- Database - базата от данни е външна за системата и се намира на отделен сървър.
- External Information Resources - този модул обединява всички външни системи, с които системата трябва да прави връзка. Например: Система за управление на учебното съдържание, Държавни публични регистри за текущи студенти, Система за контрол на национална агенция за приходите и данъчната администрация. (списъкът с външни системи, може да се увеличи в процеса на използване на системата)

с) Подробно описание на всеки модул

1. Клиентска част

1.1 GUI(Графичен потребителски интерфейс) - отговаря за представянето на съдържанието на системата при клиента и помага на потребителите да разберат и усвоят системата.

1.1.1 Registration Form - форма за регистриране на нов потребител в системата. До тази функционслност имат достъп само администраторът, членовете на административния отдел.

- private bool verificateKEY(int KEY);
- pubic bool registrate(string username, string password, int KEY);
- private bool checkKEY(int KEY);
- private bool submitRForm (string username, string password, int KEY);

  - Входни данни: KEY, след което се подават лични данни на нов потребител
  - Резултат: регистриране на нов потребител в системата
  - Грешки: при невалидна заявка или грешен KEY се връща подходящо съобщение
  - Зависимости от други елементи: след верифициране заявката се изпраща към базата данни

1.1.2 Login Form - форма за вписване на съществуващ потребител в системата.

1.1.2.1 Log in - вход в системата.

- pubic bool login(string username, string password, int ID);
- private bool checkUsername(int username);
- private bool checkPassword(int password);
- private bool checkID(int ID);
- private bool submitLForm (string username, string password, int ID);

  - Входни данни: име, парола и ID
  - Резултат: влизане на потребителя в системата
  - Грешки: при невалидна заявка се връща подходящо съобщение
  - Зависимости от други елементи: след верифициране заявката се изпраща към базата данни

1.1.2.2 Log out - прекратява потребителска сесия.

1.1.3 Client Information - служи за визуализация на основната информация на логнатия потребител, както и за предоставяне на интерфейс,

чрез който потребителят да може да достъпва своята информация и функционалности.

1.2 Validation - модул, който отговаря за валидиране на различни входни данни, подадени от клиента още преди да се изпратят към сървъра.

2. Сървърна част - модул, който се грижи за бизнес логиката. Той включва в себе си подмодули, които обособяват отделни функционалности на системата от гледна точка на клиентите.

2.1 Account Manager - този модул събира функционалността на системата, свързана с автентикация и оторизация на потребителите като предоставя подходящ потребителски интерфейс.

2.1.1 Users

2.1.1.1 Academic Department - потребителите от този отдел приемат предложения за учебни планове и програми от преподавателите.

2.1.1.2 Student Council - потребителите от студентския съвет могат да създават заявки за различни искания.

2.1.1.3 Accounting Department - потребителите от този отдел контролират финансовите операции, които засягат другите потребители (например: студентски такси, възнаграждения на служителите) и разплащания с външни изпълнители на услуги.

2.1.1.4 Administration Department - преглеждат и одобряват заявки за различни искания от преподавателите и студентския съвет. Също така одобряват предложения за учебни планове и програми.

2.1.1.5 Students - потребителите имат възможност да записват одобрени курсове, които са в рамките на тяхната специалност и при условие, че профилът им отговаря на изискванията за съответния курс. Имат достъп до електронна студентска книжка в своя профил. Също така могат да генерират различни видове официални справки за студентския си статус (системата предоставя API за достъп до генерираните официални справки)

2.1.1.6 Teachers - преподавателите могат да създават заявки за различни искания, както и да нанасят оценки в електронните книжки на студентите. Също така имат функционалност да предлагат учебни планове и програми.

2.1.1.7 Admin - модул, който обособява функционалностите на администратора на системата (поддръжка и планирано обновяване). Error handling - този модул е за отказоустойчивостта на системата. Когато има критични събития се изпраща уведомление(имейл) до администратора на системата. Системата да може да издържа на пикови

натоварвания, като може да обработва едновременно 1000 заявки в секунда.

2.1.2 Creating events - системата поддържа механизъм за публикуване на публични събития, които да може да се създават от всички потребители. (системата предоставя API за достъп до публичните събития)

2.1.3 Messages - всички потребители могат да обменят лични съобщения.

- public string searchUser(string user);
- private bool checkUser(string user);
- public void createMessage(string message);
- public bool send(string message);
- private bool submitToDB(string message);
- private void displayMessage(string message);

  - Входни данни: търсене на user и съобщение
  - Резултат: изпращане на съобщение до друг потребител
  - Грешки: при невалидна заявка се връща подходящо съобщение
  - Зависимости от други елементи: след верифициране заявката се изпраща към базата данни

2.2 Security - модул, който осигурява защита на всички лични и финансови данни от неоторизиран достъп.

2.2.1 Clearance level manager - този модул е за дефинирането на различни видове потребители и тяхното ниво на достъп до функционалностите платформата. (например : администратор - достъп до всичко и нормален потребител - ограничен достъп).

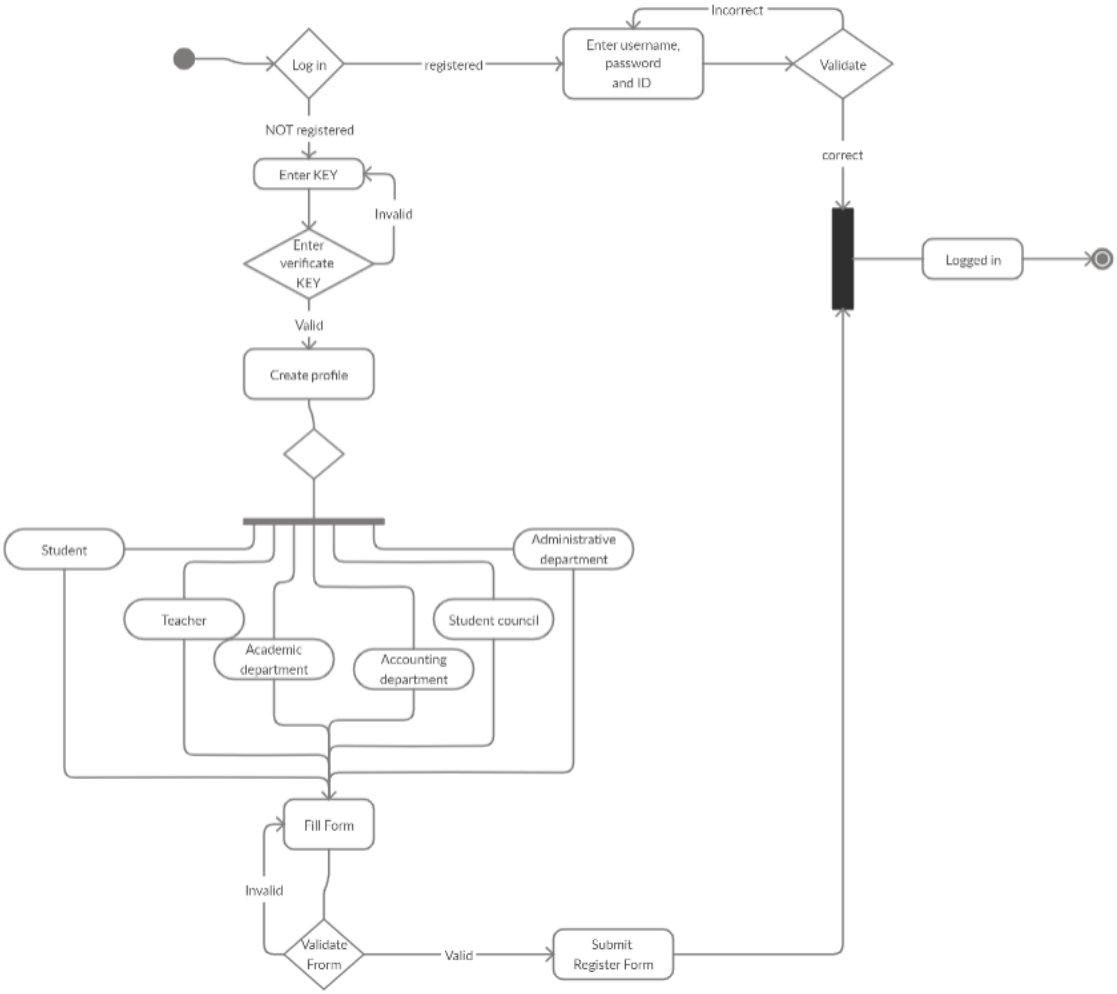
2.2.2 Log manager - този модул е с цел предотвратяване на неправомерни действия.

d) Описание на възможните вариации

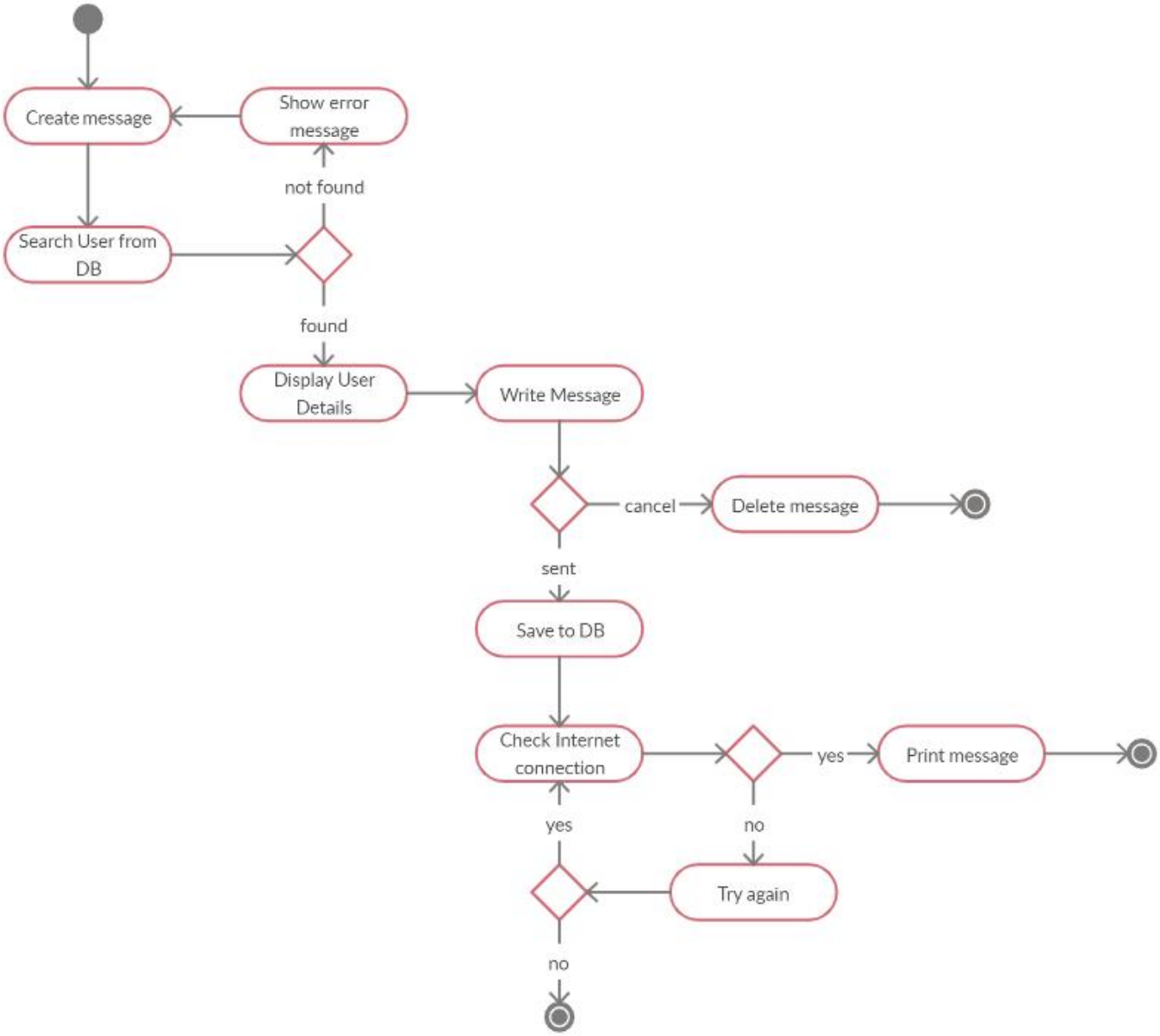
Възможна вариация на декомпозицията на модули е отделянето на модула Admin извън модула Server.
Друга вариация е да добавим в модула Account Management модул Academic Calendar, който ще е с подходящ интерфейс. Той ще служи за подреждането на различните създадени събития в структура с подходящ изглед. Потребителят ще се ориентира по-лесно за предстоящи и изминали събития.

3. Описание на допълнителните структури

3.1. Структура на процесите
а) Първично представяне



Фиг.1



Фиг.2

b) Описание на елементите и връзките

за Фиг.1

Потребителят на системата се намира в UniDigit. Ако той вече е регистриран - въвежда име, парола и ID - това е съответен идентификационен номер (например ако е студент въвежда факултетен номер, а другите типове потребители си имат уникален номер, генериран от системата при регистрация). Потребителят получава валидация от базата данни и влиза в профила си, ако е въвел правилни данни. Ако валидацията не се осъществи, той ще се върне в началото на процеса.

Ако потребителят няма регистрация, е длъжен да въведе уникален KEY, който е известен само на хората от Админ, Административния и Учебния отдел, защото само те могат да правят нови акаунти. Ако валидацията на KEY-а не се осъществи, потребителят ще трябва наново да въведе KEY-а. Ако той получи валидация от базата данни(въвел е правилния KEY) - сайтът го препраща към страница за създаване на профил.

Той може да избере типа на профила, който иска да създаде – за член на: Студентски съвет, Учебен отдел, Административен отдел, Счетоводен отдел, за студент или за учител. Във формата за регистрация трябва да се попълнят съответните данни(име, презиме, фамилия, имейл, уникален номер (за студентите е факултетния им такъв) и т.н.). Формата се валидира от системата. Ако валидацията не завърши успешно, потребителят ще бъде върнат в страницата за попълване на формата. Ако валидацията е успешна - потребителят е направил регистрация в сайта. Съответно вече може да се влиза в новонапревения акаунт.

за Фиг.2

Регистриран потребител, който е влязъл в акаунта си, иска да изпрати съобщение на друг потребител.
Избира да направи съобщение, след което вписва данни за търсене на съответния потребител-получател. Ако тези данни не съвпадат с потребител от системата, се извежда грешка и потребителят ще трябва наново да въведе данни за търсене. Ако има съвпадение на подадените данни с тези в базата данни на потребителя се показва списък с предложени потребители-получатели, които отговарят на въведеното. Той избира съответния потребител, на когото ще пише съобщение, и въвежда съобщението си. След това има 2 опции - да отмени изпращането, следователно се трие съобщението. Другата опция е да го изпрати. Съобщението се запазва в базата данни, след което се проверява за връзка с интернет. При наличие на такава - съобщението е изпратено успешно и се изписва на екрана и на двамата потребители. Ако няма връзка с интернет, се показва съобщение за повторен опит. Ако потребителят откаже - процесът се приключва, ако приеме - отново се проверява връзката с интернет.

c) Описание на обкръжението

За регистрацията се изисква наличието на Интернет връзка, поради нуждата от различни валидации и проверки. При създаване на съобщения се извършва връзка с базата данни(там се запазват).

d) Описание на възможните вариации

Описание на възможните вариации за Фиг.1:
Тъй като има възможност за некоректно въведените данни и в такъв случай потребителят трябва да бъде информиран. Може в самото съобщение да се казва точно къде е грешката.

Друга възможна вариация на Фиг.1 е да се избира типът на потребителя (от студентски съвет, от учебен отдел, от административен отдел, от счетоводен отдел, студент или учител) след като се попълни формата.

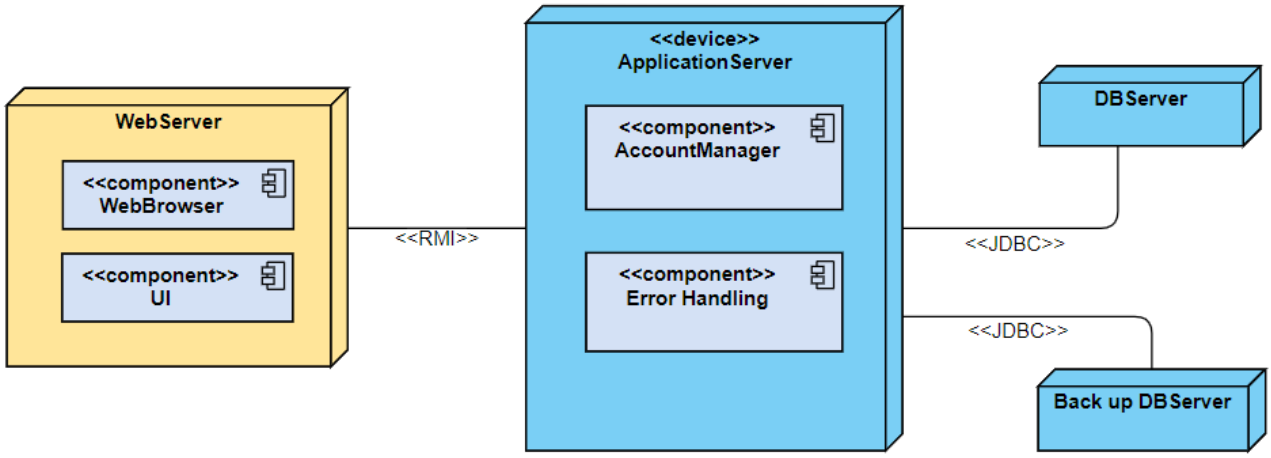
Описание на възможните вариации за Фиг.2:

На Фиг. 2 е възможно при отмяна за изпращане на съобщението да не се трие. Вместо това да се запазва в базата данни и да стои като draft (т.е. при повторно избиране на изпращане на съобщение на този потребител, предното неизпратено съобщение да е запазено).

Друга вариация е при липса на интернет връзка съобщението да не бъде успешно изпратено. Вместо това извежда съобщение за грешка,което описва възникналия проблем и има възможност да опиташ да го изпратиш повторно.

3.2. Структура на внедряването

а) Първично представяне



b) Описание на елементите и връзките

Web Server - отговаря на хардуерното устройство на потребителя на системата. В него се съдържа потребителския интерфейс (UI) на самия сайт както и Web Browser, тъй като приложението се достъпва от уеб браузър и потребителите трябва да имат такъв на своето устройство. Свързва се с Application Server посредством връзката RMI (Remote Method Invocation).

Application Server - сървър, който съдържа бизнес-логиката на системата. На него е разположено самото приложение. В него се намира Account Manager, където е логиката за различните видове акаунти и техните функционалности, и Security, където се осигурява защита на всички лични данни от неоторизиран достъп.

DBServer - на този сървър се намира цялата база от данни на системата.

Backup DBServer - важно е данните в базата да не бъдат загубени. При появата на някакъв проблем в DBServer е осигурено резервно копие на данните от нея в Backup DBServer. Този Backup DBServer е синхронизиран с DBServer базата. Application Server се свързва с DBServer чрез JDBC връзка. При неуспешно свързване с основния сървър(DBServer) се свързва с Backup DBServer.

c) Описание на обкръжението

В тази структура е по-важно разположението на самата системата, не толкова на нейното обкръжение.(затова не са показани всички връзки на системата ни с външни за нея системи.) Значително е обаче да се отбележи, че базата от данни е външна за системата и се намира на отделен сървър.

d) Описание на възможни вариации

В случай, че системата започне да се пренатоварва, възможна вариация е да се добавят дубликати на ApplicationServer, за да се обслужат част от заявките.

Друга вариация е - ако Database Backup Server не може да смогне с обработването на заявките към базата от данни - да се добавят още негови дубликати.

*И при двете възможни вариации трябва да се осигури синхронизацията, иначе ще се появят проблеми в системата.

4. Архитектурна обосновка

4.1 Функционални изисквания

4.1.1 Системата ще е разпределена на 4 приложения- Client, Server, External Information Resourses, Database.

Client- web сайт, който дава възможност на крайния потребител за управление на профила, регистрация(ако има това право) и влизане в системата.

Server – приложение, което изпълнява бизнес логиката на системата - предоставя възможност за управление на процесите и студентската информация в един университет.

External Information Resourses - външни системи, с които се свързва UniDigit. Те трябва да обменят и сихронизират информацията за университета. Подпомагат цялостта и работата на системата.

Database - база данни, която се грижи се за съхраняването на данните и тяхното възстановяване, когато е нужно.

4.1.2 В Сървърът се извършват комуникациите с външни системи. Там е и защитата на личните данни, за да се предотврати неоторизиран достъп. Освен това предоставя на различните потребители да правят-разплащания, нанасяне и преглеждане на оценки, създване на събития, съобщения, заявки за документи и други процеси, които зависят от вида потребител.

4.1.3 Системата трябва да има опростен и интуитивен графичен интерфейс за по-лесно разбиране и използване от крайните потребители.

4.1.4 Системата функционира, когато има връзка с интернет.

4.2 Нефункционални изисквания

4.2.1 За отказоустойчивостта на системата се използва тактиката активен излишък. Има дубликат (Backup DB) на основната база данни. Backup DB е синхронизирана с Database. При възникване на отказ, системата се обръща към BackUp DB.

4.2.2 За сигурността на системата има автентификация на данните – идентификационен номер (ID), който е факултетен номер за студентите и уникално генериран за останалите. Освен това има и оторизация на данните, защото всеки различен потребител има достъп до специфични за своята роля функционалности. За ограничаване на достъпа при регистрация е използван KEY, които позволява започване на процеса, само ако съответства с ID на някой от административния, учебния отдел или от админа. При засичане на опит за направа на профил от външно устройство(такова, което не е от университета) се изпраща сигнал до админа.

4.2.3 За изменяемост на системата е използвана тактика за локализиране на промените т.е. модулите са разделени, така че обхватът на очакваните промени да бъде ограничен. В допълнение има и предположения за най-вероятни бъдещи промени и при необходимост от преправяне как би било възможно да се предотврати засягането на повече модули.