

СОФИЙСКИ УНИВЕРСИТЕТ

“СВ. КЛИМЕНТ ОХРИДСКИ”

Факултет по математика и информатика

специалност Информатика

специализация Информационни и комуникационни технологии

ДИПЛОМНА РАБОТА

Тема:

WEB БАЗИРАНО ПРЕДСТАВЯНЕ НА ТРАДИЦИИТЕ И ОБИЧАИТЕ НА БЪЛГАРИЯ

Дипломант:

Надежда Каменова Иванова

Факултетен № 42 280

Научен ръководител:

ст. ас. Евгения Ковачева

Съдържание

1.УВОД.....	3
2.ЦЕЛ И ОБЗОР НА ПРОБЛЕМНАТА ОБЛАСТ.....	4
2.1.ЦЕЛ НА ДИПЛОМНАТА РАБОТА.....	4
2.2.ОБЗОР НА ПРОБЛЕМНАТА ОБЛАСТ	5
2.2.1. СРАВНИТЕЛНА ХАРАКТЕРИСТИКА С ДРУГИ МЕЖДУНАРОДНИ САЙТОВЕ.....	6
3.ИЗБОР НА СОФТУЕРА ЗА РАЗРАБОТКА НА САЙТА.	10
3.1.КАКВО ПРЕДСТАВЛЯВА PHP И ЗАЩО ГО ИЗБРАХМЕ?.....	11
3.2. ЗАЩО ИЗБРАХМЕ MYSQL?.....	16
3.3. ЗАЩО ИЗБРАХМЕ FLASH ЗА СЪЗДАВАНЕ НА АНИМАЦИЯТА ?	16
4.ПРОЕКТИРАНЕ НА РЕШЕНИЕТО.....	17
4.1.ПРОЕКТИРАНЕ НА СТРУКТУРАТА НА САЙТА.....	18
4.2.ПРОЕКТИРАНЕ НА СТРУКТУРАТА НА ФОРУМА.....	20
4.2.1.КАКВА ФУНКЦИОНАЛНОСТ ТРЯБВА ДА ИМА ФОРУМА?	20
4.2.2.ОСНОВИ НА БАЗА ДАННИ (БД).	22
4.2.3.ДЕФИНИРАНЕ НА ТАБЛИЦИТЕ ЗА БД.....	27
4.3.ПРОЕКТИРАНЕ НА FLASH КЛИПА.	36
5.РЕАЛИЗАЦИЯ НА ФОРУМА.	37
5.1.ПРЕГЛЕД НА ИНТЕРФЕЙСА И ФУНКЦИОНАЛНОСТТА.	38
5.1.1.СТРАНИЦИ ДОСТЪПНИ ЗА ОБИКНОВЕНИЯ ПОТРЕБИТЕЛ.	38
5.1.2.СТРАНИЦИ ДОСТЪПНИ ЗА АДМИНИСТРАТОРА.	44
5.2.РЕАЛИЗАЦИЯ НА ФУНКЦИОНАЛНОСТТА.	46
6.РЕАЛИЗАЦИЯ НА FLASH АНИМАЦИЯТА.	59
7.ТЕСТВАНЕ, ОЦЕНКА И УСЪВЪРШЕНСТВАНЕ НА САЙТА.....	64
8.ЗАКЛЮЧЕНИЕ.	66
9.ИЗПОЛЗВАНА ЛИТЕРАТУРА.....	67
10.ПРИЛОЖЕНИЯ.....	68
10.1.ПРИЛОЖЕНИЕ 1.	68
10.2.ПРИЛОЖЕНИЕ 2.	70

1. Увод

С развитието на информационните технологии през последните десетилетия компютърът зае своето неотменно място в обществото. Навлязоха нови гъвкави методи за визуализиране и обработка на използваната информация. Интернет предоставя възможност да се получава бърз. достъп до данни, даващи пълна информация на клиента, без ограничения във времето. Това би спестило досадното лутане между справочници, помагала, учебници. Днес Web-сайтове се очаква да съдържат много информация и да имат добро представяне, но не и да разсейват с труден и тромав интерфейс. Информацията в него трябва да е лесно откриваема и актуална. Един ясен и динамичен Web сайт е голяма ценност за потребителя, който иска бързо и лесно да достигне до желаната информация.

Целта на настоящата дипломна работа е да се разработи Web сайт, предлагащ изчерпателна информация за автентичните българските обичаи и традиции, така, както са били чествани през миналите столетия до средата на XX век. Тъй като интересът на чуждестранните граждани към страната ни нараства значително през последните години основна цел на дипломната работа е този сайт да бъде достъпен и за тях. По тази причина сайтът е разработен както на български, така и на английски език.

Дипломната работа може условно да бъде разделена на две части – динамична и статична. Сайтът е разработен от екип от две дипломантки и по тази причина описанието на разработката е разделено на две части:

- статична част, обхващаща съдържанието на WEB-сайта, обзор на проблемната област, оформление, структурата и йерархията;
- динамична част, реализираща взаимодействието с потребителите – проектирането и разработването на форум, даващ възможност на потенциалните читатели на сайта да обменят мнения.

Тъй като дипломната работа е разработена в екип с Камелия Минчева Косева, и двете части ще бъдат разгледани в настоящата дипломна работа, но по-подробно ще бъде описан само от една от двете дипломантки. В настоящата дипломна работа повече внимание ще бъде обърнато на динамичната част на сайта. Статичната част на сайта и на неговото дизайнерско оформление ще бъде

разгледана с повече подробности в дипломната работа на Камелия Минчева Косева.

2.Цел и обзор на проблемната област

2.1.Цел на дипломната работа

Настоящата дипломна работа има за цел разработването на Web сайт, съдържащ информация за традициите в празнуването на някогашните обичаи и обреди на българите. Това включва по-значимите църковни и народни празници, които са разделени според народния български календар. Предназначението на тази дипломна работа е да обхване колкото е възможно повече материали по темата, да ги систематизира и да създаде Web сайт, който да е атрактивен за потребителите.

Информацията в сайта трябва да е поднесена по неангажиращ и увлекателен начин, достъпна за широк кръг от потребители. Обичаите трябва да са представени както според църковния календар, така и според езическите вярвания на българите. Текстът следва да е богат в стилистично отношение и да бъде поднесен на български и английски език, тъй като стремежът е сайта да спечели не само български потребители, но и такива от цял свят. Трябва да бъде поднесена и графична информация, която да допълва описанието на българските традиции. Трябва да има място на сайта, на което потребителите да могат да обменят информация, а именно форум.

Преди да започнем работа по нашето приложение трябваше да го разделим на няколко основни компоненти и да определим как ще работи всеки един от тях, и как те ще се съвместят заедно.

Въз основа на поставената по –горе цел се оформиха следните компоненти:

- Изследване на източниците на информация за българските обичаи и традиции, въз основа, на които ще бъде построен сайта. Това включва проучване на наличната информация в Интернет пространството с

подобно съдържание, както и на съществуващите книжни източници, описващи народните обичаи и традиции;

- Проектиране на решението. Какви са целите на сайта и как трябва да бъде организиран?
- Избор на софтуер за разработка на сайта.
- Проектиране на форума. Проектиране на базата данни, която ще обслужва форума.
- Реализация на графичния интерфейс на сайта. Подредба на менютата, графичните ресурси, форума и статичната информация.
- На базата на по –горните задачи да се създаде окончателният вариант на сайта.
- Тестове, оценката и усъвършенстване на сайта - реалните постижения на изготвеният сайт, неговите недостатъци и предложения за бъдещо усъвършенстване и развитие.

Настоящият сайт е разработен в екип с Камелия Косева , затова изложените по-горе задачи са разделени в две дипломни работи, всяка обхващаща част от тях. В дипломната работа на Камелия Косева е описана статичната част на сайта, а настоящата дипломна работа описва динамичната част на сайта.

2.2.Обзор на проблемната област

При създаването на сайта за българските традиции първо направихме проучване на българското и чуждестранното интернет пространство за съществуващи сайтове на подобна тематика - как са организирани тези сайтове, какъв софтуер е използван за тяхната реализация, до колко тези сайтове са изчерпателни, по какъв начин е поднесена информацията.

Необходимостта от токова проучване възникна, за да може да се очертае нужния модел за разработката и за да се проучи какво потребителя интуитивно очаква да види в подобен род сайтове. Дефинирахме следните критерии за оценяване на подобни Web –сайтове:

- **Въведение** - разглеждане на заглавната част на даден сайт, до колко тя е атрактивна и увлекателна, до колко може да задържи вниманието на потребителя.

- **Определяне потребителите на дадения сайт** – кой би използвал дадения сайт.
- **Интерфейс на сайтът** - как изглежда сайтът графично, до колко е атрактивен или претрупан, интуитивна ли е ориентацията по страниците.
- **Изчерпателност и организация на информацията** - каква информация съдържа дадения сайт и до колко тя е покрила засегнатата област и по какъв начин е структурирана.
- **Навигация** - до колко лесно и интуитивно става предвиждането по страниците на сайта.
- **Скорост на зареждане** - с каква скорост се зарежда сайта, до колко са оптимизирани графичните ресурси.
- **Софтуер** - какъв софтуер е използван за изработването на сайта.
- **Динамичност на сайта** - има ли възможност потребителя да общува със сайта.

По тези критерии е направен анализ на съществуващите подобни сайтове в българското Web пространство в дипломната работа на Камелия Минчева Косева. В настоящата дипломна работа ще бъде направен анализ на съществуващите подобни сайтове в международното Web пространство.

2.2.1. Сравнителна характеристика с други международни сайтове.

Търсейки в мрежата подобни по тематика Web сайтове установихме, че съществуват много малко, които да предлагат подробна и систематизирана по съответния народен календар информация за традициите на другите народи. Повечето страни са посветени на ограничен брой празници. Съществуват много сайтове, които са посветени на празнуването на сватбените традиции, Великденските и коледни празници в различните страни. Тяхно предимство е, че информацията е достъпна от едно място и потребителят може лесно да сравни начина на празнуването на празниците за отделните страни. Примерни сайтове са <http://www.christmas.com/>, <http://www.holidays.net/easter> , <http://www.geocities.com/Paris/LeftBank/3852/easter.html> и

<http://www.worldweddingtraditions.com>. Това, което не ни допадна в тези страници е, че са претрупани с много реклами отвличащи вниманието на посетителя и не са изцяло ориентирани към проучването на дадения обичай. Голяма част от информацията за празниците на другите страни открихме предимно в туристически web сайтове, но тя не е изчерпателна .

Обща черта на повечето от сайтове, посветени на традициите е, че те не предлагат никакъв начин за комуникация с посетителите. Много малко от тях са атрактивни и предлагат достатъчен графичен материал за описанието на обичаите. Тук ще направим сравнение само с няколко сайта, тъй като не е възможно това да се направи за всички, които открихме.

Румънски обичаи и традиции

http://www.ici.ro/romania/en/traditii/cust_home.html

- **Въведение** - Заглавната страница е без ефекти и предлага кратък увод относно съдържанието на сайта.
- **Определяне потребителите на дадения сайт** – Сайтът е достъпен за всички англо- говорещи потребители.
- **Интерфейс на сайта** - Сайтът е семпъл, няма вграден специфичен елемент, който да отличи румънските обичаи от такива на други народи. Менюто е разположено в дясно и има препратки към съответните празници, разделени по сезони.
- **Изчерпателност и организация на информация** - информацията според нас е сравнително пълна, има разделение по сезони на празниците, както и информация за специфични румънски обичаи, като “Игри с маски”. Има достатъчно снимков материал, които да допълва описанието на обичаите.
- **Навигация** - Навигацията е затруднена. Това, което не ни допадна е, че отивайки на даден обичай, главното меню изчезва и няма начин посетителя да се върне на главната страница освен чрез Back бутона на брауъра. По този начин ориентирането на потребителя в

страниците на сайта е по – трудно. Липсва карта на сайта, която да очертава, каква информация е включена.

- **Скорост на зареждане** - Снимковият материал не е оптимизиран и страниците, на които той съществува, се зареждат значително по-бавно.
- **Софтуер** - Използван е само HTML.
- **Динамичност на сайта** – Сайтът е статичен. Потребителят няма възможност да взаимодейства със сайта по никакъв начин.

Английски обичаи и традиции

http://www.geocities.com/traditions_uk/index.html

- **Въведение** - Подобно на предишният разгледан сайт и тук заглавната страница съдържа само кратко въведение относно тематиката на сайта.
- **Интерфейс на сайта** - Сайтът е изчистен и липсват често срещаните реклами, отвличащи вниманието на потребителя. Идеята за карта на сайта, която да улеснява допълнително потребителя, която е направена в настоящия сайт е заимствана от този сайт.
- **Изчерпателност и организация на информация** - информацията според нас е сравнително пълна и е разделена по месеци. Има информация за произхода на всеки от описаните обичаи. Като недостатък на сайта може да се отбележи липсата на графичен материал, който да допълва описанието на традициите.
- **Навигация** - Както и в предишния описан сайт и тук навигацията не е реализирана по най – елегантния начин. Липсата на меню, което да улеснява придвижването по страниците е недостатък на сайта . Отваряйки страницата на даден обичай потребителя не може да се върне в главната страница освен чрез Back бутона на брауъра.
- **Скорост на зареждане** - Тъй като няма графични елементи скоростта на зареждане е бърза.

- **Софтуер** - Използван е само HTML.
- **Динамичност на сайта** – Потребителя има възможност да даде своя коментар за сайта и да изпрати информация за съществуването на този сайт на свой приятел посредством HTML форма.

Унгарски обичаи и традиции

<http://www.hungarotips.com/customs/>

- **Въведение** - Въведението в сайта е аналогично на разгледаните по – горе страници.
- **Интерфейс на сайта** - Сайтът е семпъл и предлага снимков материал описващ част от празниците. Страниците в сайта не са оформени в единен стил - заглавната страница и страниците съдържащи информация за отделните обичаи са с различна цвятова гама.
- **Изчерпателност и организация на информация** - информацията според нас е непълна. Една от добрите черти на сайта е, че има подробна карта на именните дни. За някои от обичаите съществува снимков материал. Едно от предимствата на този сайт са предложените игри (пъзел и тестове), които биха заинтригували потенциалните потребители.
- **Навигация** - Ориентирането на потребителите в страниците на сайта е затруднено, тъй като не съществува меню, което да улеснява придвижването по страниците. Отивайки на даден обичай потребителя няма да се върне на главната страница освен чрез Back бутон на браузъра. Също някои от препратките на сайта сочат към не съществуващи страници.
- **Софтуер** - Използван е HTML и JavaScript

В интернет пространството съществуват още много сайтове, съдържащи подобна информация за някоя страна, но в тях се описват само малка част от обичаите. Не претендираме, че познаваме обичаите на всички народи, но е

странно, когато за една страна са описани само ограничен брой традиции. В повечето случаи темата не е засегната достатъчно задълбочено и подробно. По-голяма част от тези сайтове не са проектирани, така че да улесняват потребителя – навигацията в тях е лоша, за читателя е трудно да открие нужната информация, придвижването в тях е затруднено и е почти невъзможно за потребителя да проследи откъде е минал и как да се върне на мястото, от което е тръгнал.

В сравнение с разгледаните български сайтове, съдържащи информация за народните ни обичаи, наличните сайтове описващи традициите на други страни предлагат значително по – оскъдни сведения, информацията в тях не е систематизирана и достъпът до нея е затруднен поради лошата навигация.

Вземайки предвид всички тези негативни страни на съществуващите сайтове за българските обичаи и традициите на други страни, ние се постарахме да създадем по-богат, с добра навигация и илюстриран с много изображения сайт за българските обичаи.

3.Избор на софтуера за разработка на сайта.

Използването на HTML (Hypertext Markup Language) дава възможност на визуализиране на текст и графики в Web браузъра. Но сайт изграден само със помощта на чист HTML притежава много ограничения. Статичното съдържание на такъв сайт си остава винаги едно и също, освен ако не се обновява физически. Съвременните сайтове трябва да съдържат в себе си много информация, която да бъде добре представена от графична гледна точка. Няма да бъдат разглеждани HTML технологията, JavaScript и каскадните стилове (CSS), защото те са описани в дипломната работа на Камелия Минчева Косева.

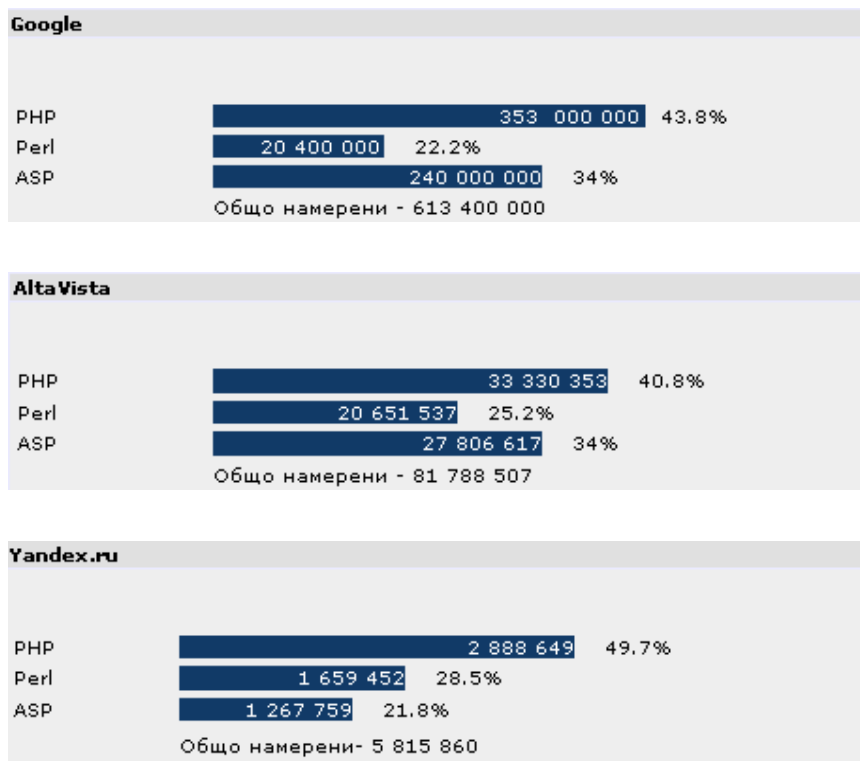
Динамичните web-сайтове днес не са просто Web страници – те включват още начини за съхранение на данни и заявки към тях (възможно е с бази данни), начини за обработка на заявките от потребителя и създаване на документи с подходящата информация.

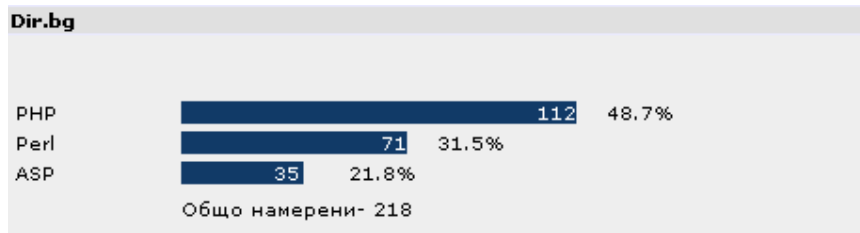
Сайтът за българските традиции ще е динамичен, тъй като потребителя ще има възможност да комуникира със сайта посредством форум. Тоест необходим е софтуер, с които да можем изградим тази динамичност. За изграждането на нашия сайт избрахме като програмен език PHP, а като бази данни MySQL.

3.1.Какво представлява PHP и защо го избрахме?

PHP е сървърен скрипт език, проектиран специално за web нужди. PHP кодът може да се вгради в HTML страница и се изпълнява при всяко нейно посещение. Кодът се интерпретира от Web сървъра и генерира HTML или други изходни данни, които посетителя на сайта вижда. Чрез изпълняване на PHP програми на сървъра, може да се създават много мощни приложения, които си взаимодействат с базата данни и динамично генерират съдържание.

Защо не избрахме технологиите ASP (Microsoft Active Server Pages), Perl или JSP (Java Server Pages)? Отговорът е: простота, почти естествен начин за използване на бази данни и независимост на платформата. Също така направихме проучване кой е най – използвания език в Интернет. Осъществихме това изследване чрез най – популярните търсачки в Интернет. Разсъжденията бяха такива: колкото повече се обсъжда даден език, толкова по – интересен е той за читателите, колкото по често се среща името на езика, толкова по – популярен е той. Ето какви резултати получихме (фиг.1) по количеството намерени изрази:





Фиг. 1. Популярност на PHP спрямо езиките Perl и ASP

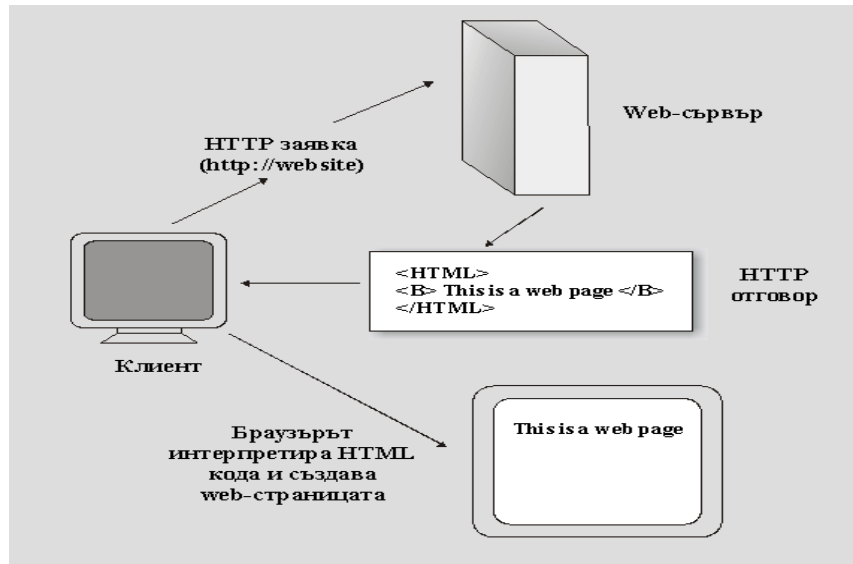
Горният резултат според нас се корени от следните предимства на PHP:

- **PHP е безплатен** - PHP е софтуер с отворен код. Това означава, че няма ограничения в целите и предназначенията, за които се използва, както и за хората, които имат право да го използват. Потребителят може да прави и разпространява произволни модификации, заедно с оригиналния софтуер.
- **PHP е лесен** - PHP не изисква задълбочено разбиране на основен език за програмиране. Много от най-полезните функции са предварително дефинирани.
- **PHP е вграден** - PHP е вграден в HTML. Това означава, че страниците на PHP са обикновено страници, които “превключват” в режим PHP, когато е необходимо. Това от своя страна има много полезни последствия: PHP може бързо да бъде добавен към кода, създаден от някой редактор; PHP води до разпределяне на работата между дизайнерите и авторите на скриптове; не е нужно всеки ред на HTML да бъде писан отново на език за програмиране; PHP може да намали цената на разработването и да увеличи ефективността; не е необходимо PHP да се компилира, той се интерпретира.
- **PHP е между платформен** - PHP работи на всеки популярен вариант на UNIX или Windows. Огромен процент от сървърите за HTTP по света работят на един от тези два класа операционни системи. PHP е съвместим с трите водещи сървъра за Web: Apache HTTP Server за UNIX и Windows, Microsoft Internet Information Server и Netscape Enterprise Server. Тук ASP губи, защото не се поддържа от сървър Apache - най популярният сървър не само в България.

- **PHP е независим** - PHP не е привързан към нито една от операционните системи за сървъри. Той не е свързан и към нито един от собствените между платформени стандарти и към нито един браузър или реализация на програмен език, или база от данни.
- **PHP улеснява комуникирането с други програми и протоколи** - екипът по разработката на PHP се е заел с осигуряването на максимална гъвкавост за възможно най-голям брой потребители. PHP притежава вградена поддръжка на драйвери за приблизително 15 от най-популярните бази от данни. PHP поддържа и голям брой основни протоколи като POP3 (Post Office Protocol Version 3 - протокол за изпращане и приемане на съобщения), IMAP (Interactive Mail Access Protocol - протокол за изпращане и приемане на съобщения), LDAP (Light-weight Directory Access Protocol –протокол за достъп информация, свързана с директория), SMTP (Simple Mail Transport Protocol - протокол за изпращане на поща по Интернет) и други.
- **PHP се интегрира лесно с бази данни** – PHP притежава вградени възможности за връзка с множество бази данни като MySQL, PostgreSQL, mSQL, Oracle, InterBase, Sybase и други. Чрез използване на стандарта Open Database Connectivity (ODBC), PHP може да се свърже с всякаква база данни, притежаваща ODBC драйвер.

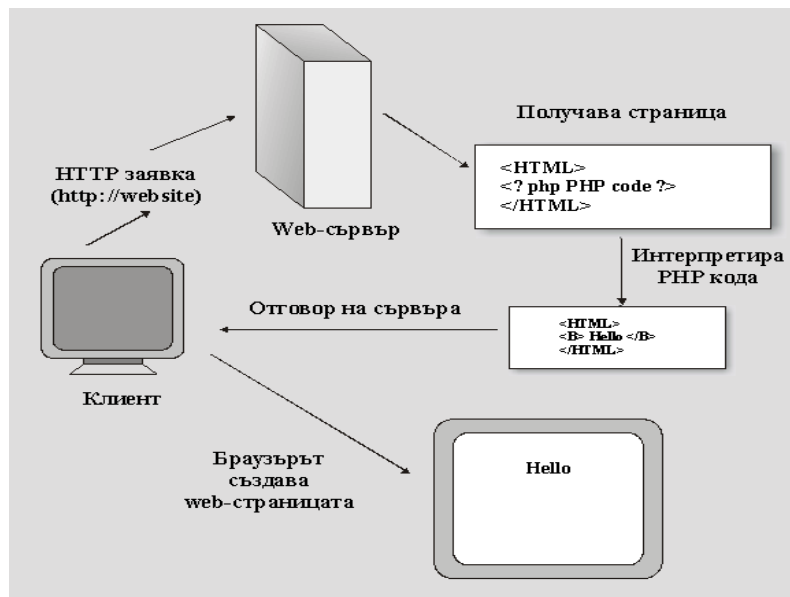
Основното различие между PHP и HTML страниците е в начина , по който работи с тях Web-сървъра.

- **Какво става с HTML страниците?** - когато се получи заявка от браузъра за дадена страница, Web-сървърът изпълнява три стъпки (фиг. 2):
 1. Прочита заявката на браузъра;
 2. Намира исканата страница на сървъра.
 3. Изпраща обратно тази страница чрез Интернет към браузъра.



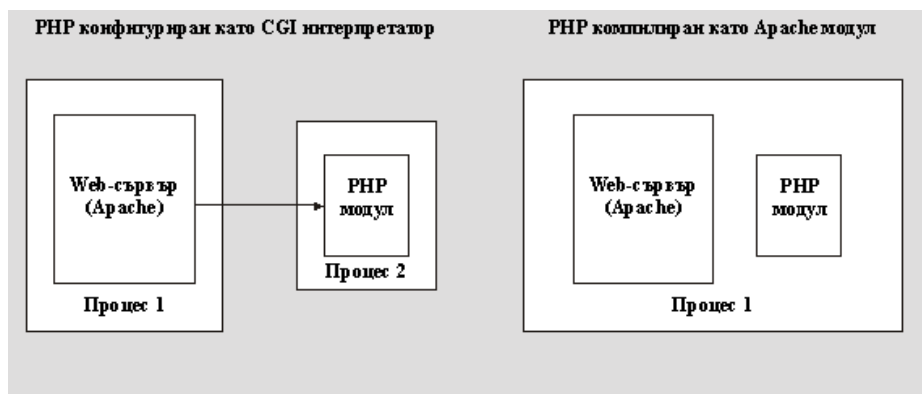
Фиг.2. Взаимодействие на HTML страниците със сървъра

- **Какво става с PHP страниците?** - вместо да се изпраща към потребителя статична HTML страница, искането е сървъра да извърши определени действия в зависимост от потребителския PHP код: PHP ще вземе някои решения и ще създаде страница, която е подходяща за конкретна ситуация. Така, че когато се използва PHP, действията на сървъра са следните (фиг. 3):
 1. Прочита заявката от брауъра.
 2. Намира исканата страница на сървъра.
 3. Изпълнява инструкциите, заложиени в PHP, за да модифицира страницата.
 4. Изпраща страницата обратно чрез Интернет към брауъра.



Фиг. 3. Взаимодействие на PHP страниците със сървъра

PHP може да бъде конфигуриран и компилиран по два начина - като самостоятелен интерпретатор и като Apache модул. Когато PHP е конфигуриран като CGI(Common Gateway Interface) интерпретатор, всеки път когато PHP скриптът се интерпретира, Web сървърът създава инстанция на PHP интерпретатора, която интерпретира скрипта. Това обаче влошава представянето. Когато PHP е компилиран като Apache модул (фиг. 4.), той се изпълнява в същото адресно пространство като това на процеса на Web сървъра, което осигурява подобряване на представянето в сравнение с традиционните CGI интерпретатори, които са отделни процеси. Постоянна връзка с база данни е налице само във версията на Apache модула, което е основно предимство.



Фиг. 4. Схеми на компилиране на PHP(Кастането, 2001, стр.42)

3.2. Защо избрахме MySQL?

MySQL е бърза, стабилна система за управление на релационни бази данни (RDBMS) (Уелинг, 2004, стр.32). Базата данни позволява ефективното съхранение, претърсване, сортиране и извличане на данни. MySQL сървърът контролира достъпа да данните и позволява едновременната работа на множество потребители, бърз достъп, както и осигуряването на достъп само на оторизирани за това потребители. Следователно MySQL е многопотребителски, многонишков (multi-threaded) сървър. Използва се SQL (Structured Query Language –структуриран език за заявки), който е стандартният световно разпространен език за заявки към бази данни.

Основните конкуренти на MySQL са PostgreSQL, Microsoft SQL Server и Oracle. Според Уелинг (Уелинг, 2004, стр.34) предимствата на MySQL са:

- Висока производителност – бързината на MySQL е безспорна. Много от тестовете показват, че MySQL е в пъти по – бърза в сравнение с конкурентите си.
- Ниска цена – MySQL е достъпна безплатно с лиценза за отворен код, а при нужда – на ниска цена с комерсиален лиценз.
- Лесно използване – използва SQL и е лесна за настройване.
- Преносимост – MySQL може да се използва под множество UNIX системи, както и под Microsoft Windows.

3.3. Защо избрахме FLASH за създаване на анимацията ?

Macromedia Flash е софтуерно приложение, съдържащо инструменти за създаване на анимации, векторни графики, приложения, софтуер, представяния и Web сайтове. Flash публикува SWF файлове, които са икономични откъм размер и са платформено независими и могат да бъдат разглеждани посредством Flash Player. Flash Player е софтуер, използван за възпроизвеждане на SWF файлове, генерирани от Flash. SWF файловете се възпроизвеждат или директно в прозореца на брауъра или в самостоятелен прозорец наречен player, който от своя страна се

грижи съдържанието да се вижда. Тези файлове могат да бъдат видени от широка публика благодарение на голямата популярност на Flash Player, който се разпространява безплатно в Интернет, предлага се и като допълнение към много игри и други мултимедийни приложения..

Flash използва скриптов език, наречен Action Script – мощен език, базиран на ECMA Script и подобен на Java Script и Java. Въпреки, че Flash притежава скриптов език не е необходим много (или въобще никакъв) код, за да се създаде интерактивно приложение. Flash притежава инструменти, с които може да се поддържат малки размери на файловете и бързо работещи сайтове, така че да се избегне дългата фаза на зараждане на дадената страница.

Flash–филмчетата наподобяват Java applet, но за разлика от applet са много по-лесни за създаване и са по-стабилни. Applet-ите по някой път заемат много ресурси, което практически затруднява работата на брауъра. Сравнително малките по размер файлове, съдържащи качествена анимация и звук правят Flash подходящ за презентации в Web.

Flash е подходящо приложение за създаване на мултимедийни файлове и дава възможност за вмъкване на много видове медия в това число текст, графика, видео, векторни файлове, PDF файлове и аудио. По този начин позволява на web дизайнера да направи своите файлове интерактивни, динамични, забавни и интуитивни.

4.Проектиране на решението.

В настоящата дипломна работа няма да бъдат описани всички компоненти (описани в точка 2.1.), които изграждат сайта за българските традиции. Ще спрем вниманието Ви на проектиране на структурата на сайта, прилежащия му форум и уводната анимация. Проектирането на останалите компоненти ще бъде направено в дипломната работа на Камелия Минчева Косева.

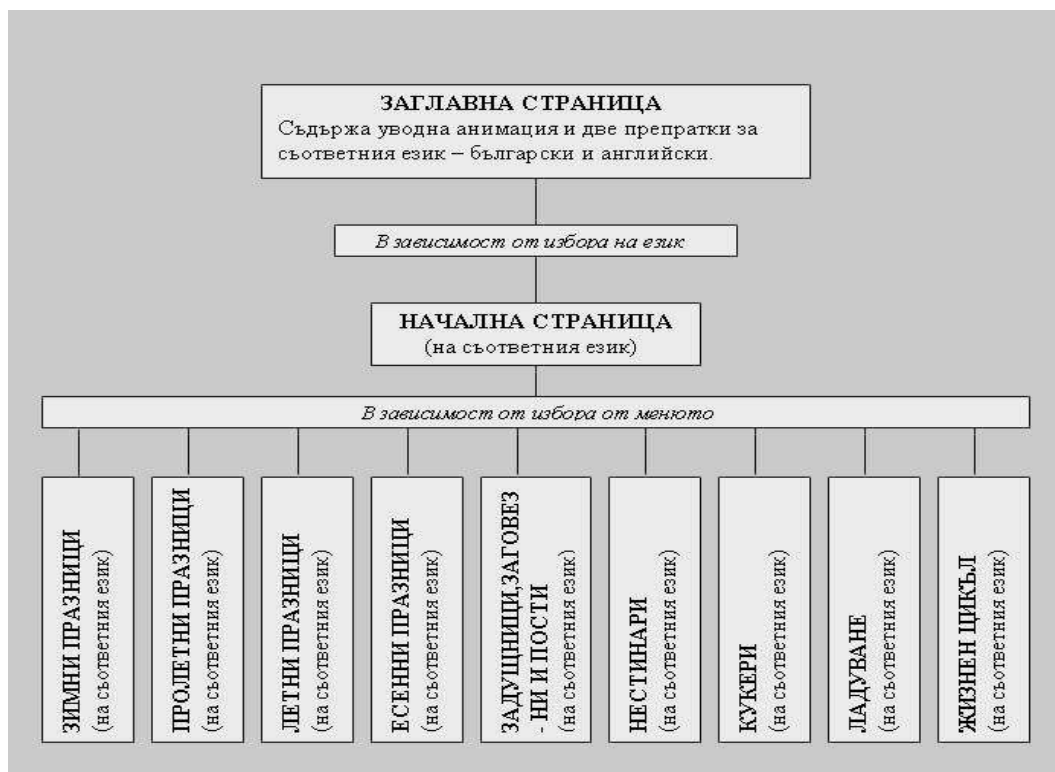
4.1. Проектиране на структурата на сайта.

Целта на сайта е да задържи вниманието на потребителя още в началото. За това ни бе нужно ефектно появяване. Решихме, че за целта ще създадем заглавна страница с анимация. От тази заглавна страница потребителя може да избере езика, на който желае да разгледа сайта. Наличните езици са български и английски. Ако посетителят желае да пропусне анимацията може да направи това, натискайки препратката за съответния език.

Информацията, с която разполагаме може да бъде групирана в няколко основни раздела, които оформят менюто. За да могат страниците да са по-малки и да се зареждат по-бързо, ще разделим допълнително информацията във всеки един от дяловете. Те ще бъдат под линкове в менюто. Основните раздели на менюто, които открихме са:

1. Зимни празници (с под линкове – Декември, Януари и Февруари).
2. Пролетни празници (с под линкове – Март, Април, Май и Великден).
3. Летни празници (с под линкове – Юни, Юли и Август).
4. Есенни празници (с под линкове – Септември, Октомври и Ноември).
5. Задушници, заговезни и пости (с под линкове – Месни заговезни, Сирни заговезни, Богородични заговезни, Задушници и Пости).
6. Нестинари (без под линкове).
7. Кукери (без под линкове).
8. Ладуване (без под линкове).
9. Жизнен цикъл (с под линкове – раждане, кръщение, сватба и погребение)

Подредбата на информацията и на двата езика е еднаква. В горната част на сайта има три препратки, съответно към заглавната страница, форума и картата на сайта. На фиг.5 е показана графически структурата на сайта.



Фиг. 5. Структура на сайта

При наличие на повече снимков материал за даден празник, той е организиран в картинна галерия. При натискане на някоя от снимките се отива в екран, който показва снимката в уголемен размер. При натискане на уголемената снимка се връщаме в общата картинна галерия.

В началото на всяка страница е указано къде се намира текущата страница в структурата на сайта. Примерно, ако се намираме в зимни празници, декември месец и разглеждаме информацията предоставена за празника Никулден, тогава в горния край на екрана, веднага под менюто е указано: **зимни празници: декември: Никулден - 6 декември.**

Навигацията на сайта е реализирана чрез разгъващо се меню. То е достъпно от всяка страница на сайта и по този начин предвижването по страниците става лесно и интуитивно. В долната част на всяка страница има две препратки - **назад** и **нагоре**, които съответно ни връщат на предишната страница и в началото на текущата.

Изглед на горе описаните страници може да откриете в Приложение 2 от настоящата дипломна работа.

4.2. Проектиране на структурата на форума.

Като повечето Web приложения форума се ражда от необходимостта потребителите на даден сайт да обменят идеи на дадено място.

4.2.1. Каква функционалност трябва да има форума?

Какво представлява форума? Форумът е Web приложение, което дава възможност на потребителите на даден сайт да публикуват статии или въпроси към статии, а други потребители да могат да четат и отговарят на техните въпроси. Също така потребителите имат възможност да видят по – ранни съобщения към дадена статия. Web форумите понякога се наричат още и дискуссионни бюлетини. Всяка тема на дискусия ще наричаме нишка.

Възниква вероятността не всички публикувани теми да се добронамерени. Това налага необходимост този форум да може бъде контролиран по някакъв начин. За това трябва да съществуват потребители с различни права за ползване и достъп до форума. Нивата на достъп ще са два типа – администраторско и обикновено. Администраторското ниво позволява на даден потребител да има разширени управленчески функции. Администраторът на форума може да бъде само един потребител. По – долу са посочени какви са възможностите на всеки един потребител на форума според нивата на достъп.

Форумът трябва да предлага на обикновените потребителите следната функционалност:

- Възможност да започват нови дискуссионни нишки (нови теми), като ги публикуват.
- Възможност да публикуват статии в отговор на съществуващи статии.
- Възможност да разглеждат публикуваните статии.
- Възможност да разглеждат взаимоотношението между статиите, тоест да виждат кои статии са отговор на други статии.
- Да могат да се идентифицират или не в страниците на дадена дискусия, тоест да могат да отговарят с дадено потребителско име или да останат анонимни. Тази възможност ще се реализира чрез регистрация с дадено

име и парола. При регистрацията потребителите получават уведомителни писма за регистрацията.

- При изгубване или забравяне на дадена парола да имат възможност да запазят потребителското си име и да получат нова парола.
- Регистрираните потребители да могат да получават осведомителни писма, когато е публикувано нещо ново по интересна за тях тема. А също така, ако вече дадена тема не ги интересува да спрат получаването на уведомителните писма.
- Регистрираните потребители да могат да редактират дадена своя тема или отговор.
- Възможност за търсене на статии по дадена тема и дата.
- Статистика за най – посещаваните теми и най – активните регистрирани потребители.
- Редактиране на личната информация

Форумът трябва да може да предложи на администратора му следната функционалност:

- Всички по – горе изброени функции, които са достъпни за обикновения потребител.
- Добавяне, редактиране и изтриване на форум и прилежащите му теми и отговори.
- Търсене на потребител по следните критерии: потребителско име, email, потребители без нито едно съобщение
- Забраняване достъпът на дадено IP до форума
- Разрешаване достъпът на дадено IP до форума
- Изтриване на е – mail осведомяване за даден е – mail адрес или за всички съществуващи

Ще ни трябва начин, по който да съхраняваме информацията за потребителите, форумите, темите, отговорите, е – mail-те на потребителите и техните IP адреси. За целта ще използваме MySQL база данни. Преди да преминем към структурата на таблиците ще направим малко отклонение, за да отбележим основните характеристики на релационна база данни.

4.2.2. Основи на база данни (БД).

Бази от данни представляват организирана съвкупност от взаимосвързани данни, разположени в един или няколко файла, отразяващи състоянието и измененията в наблюдавани параметри на обекти от предметната област. Те служат като единна информационна среда за приложните програми, чрез които се автоматизира процеса на управление. Данните се организират така, че да има възможност за добавяне на нови и модифициране на съществуващите, а така също и за осъществяване на успешно информационно търсене. Организацията на БД трябва да осигури минимален излишък на данните, което допуска използването им по оптимален начин за едно или няколко приложения. Данните трябва да се структурират по начин, който да предвижда възможност за по-нататъшно нарастване на приложенията им.

БД са всички сведения, които се въвеждат, съхраняват, обработват в компютър и представят на потребителя информацията в определена форма. Цялото програмно осигуряване, осигуряващо тези операции над данните, се нарича СУБД (Система за Управление на Базата Данни). Съвременните СУБД дават на потребителя възможност за автоматизирана работа с данните чрез абстрактни термини, които не са свързани с начините за физическото съхраняване на тези данни в компютъра.

В теорията на базата данни съществува стандартно понятие “ниво на абстракция на представяне на данните”. Има се предвид, че данните, от една страна, могат да се разглеждат на ниво битове, записани на външни носители, а от друга страна, тези данни представляват за потребителя съвсем други абстракции, като например имена на потребители, на продукти и др. Реално съществува само БД, разглеждана на първото физическо ниво на абстракция. Останалите нива се създават изкуствено с цел удобство на потребителя при работа с бази данни. Те са концептуални и представителни нива и се осигуряват чрез програмни средства, работещи с данните на високо ниво.

Концептуалната база данни е абстрактно изображение на физическата база данни. Може да се каже още, че концептуалната база данни се представя с една

абстракция на реалния свят, изградена върху концептуална схема. Именно такава абстракция има дълбок смисъл, тъй като базата данни съдържа информация за обкръжаващия реален свят. Всяка база данни се свързва с определена предметна област. Концептуалната схема я отразява на високо ниво, боравейки с термини като имена на потребители, заглавия на теми и форуми и др. За да се опише една такава схема, се използват специализирани езици за описание на данните. Те са езици от високо ниво и позволяват описанието на концептуалната схема на една реална база данни чрез понятията (термините) на някакъв модел данни. Утвърдили са се три модела данни:

- Мрежов модел - той е модел на ориентирани графи. В него всеки елемент може да има повече от един пораждащ елемент. Произволен елемент може да бъде свързан с произволен друг елемент. В много мрежови структури, задаващи връзките между типовете записи или типовете агрегати на данните, представянето на отношенията между изходен и породен елемент се оказва сложно, а отношението породен-изходен елемент просто. Трябва да се различават структурите, в които представянето на отношението породен-изходен елемент е просто или не се използва от структури, в които отношението между два типа данни във всички направления е сложно. Структурите от първия тип са прости, а от втория – сложни.
- Йерархичен модел - в него записите се представят във вид на дървовидна структура. Те се подразделят на йерархични категории, като всяка една категория може да се яви като подмножество на друга. Дървото се определя като крайно множество, състоящо се от възли, от които има един, специално обозначен, наречен корен. Останалите възли се съдържат в непресичащи се множества, всяко от които се явява също дърво. Те се наричат поддървета на дадения корен.
- Релационен модел - основава се на теоретично множественото понятие “отношение”. Връзки между данните са представени във формата на таблици. Всяко отношение е композиция от таблици, записи или полета. Физическият ред, под който се записват отделните

записи, практически е без значение. Всеки запис в една таблица притежава поле, съдържащо уникална стойност (полето се нарича Първичен ключ). Тези две характеристики на релационните бази данни позволяват данните да бъдат абсолютно независими от физическото им съхранение в компютрите. За един потребител не е необходимо да знае къде физически се намират данните, за да може да получи достъп до тях.

Съществуват следните термини, свързани с понятието база данни:

- **Обект** - той е съществуващ и различим елемент на реалния свят. Група от всички подобни обекти образува набор от обекти. Набори от обекти са например всички теми и отговори, всички имена на потребители и др. При създаването на конкретната база данни от голямо значение е наборите от обекти с техните характеристики да се изберат така, че да бъдат различими един от друг.
- **Атрибути** - те представляват характерните свойства на всеки един от наборите от обекти. Конкретните стойности на атрибутите, които могат да бъдат цели или дробни числа, а също така и текстови, дават възможност да се отдели един обект от набора от останалите обекти в този набор.
- **Ключ** - това е всеки атрибут или група атрибути, еднозначно идентифициращи всеки обект в набора от обекти. Всеки набор трябва да има ключ, който трябва да бъде определен така, че даденият набор да се отличава от всички други набори от обекти. Липсата на ключ сред атрибутите на даден набор обекти не дава възможност да се отличи даден обект от останалите.
- **Връзки** - връзките между наборите от обекти представляват просто подреден списък от набори от обекти. Конкретен набор от обекти може да се появява в този списък само веднъж.

Видове функционални зависимости - в теорията на базите данни връзките се класифицират поради това, че в общия случай е възможно определено количество

обекти от даден набор да бъдат свързани с определено количество обекти от друг набор. Различават се следните видове връзки:

- **1:1 (едно-към-едно)** - на едно поле от таблица А отговаря точно едно поле от таблица Б и обратното;
- **1:N (едно-към-много)** - на едно поле от таблица А отговаря едно или няколко полета от таблица Б. На едно поле от таблица Б отговаря точно едно поле от таблица А. Подобна връзка се постига чрез добавяне на ново поле в таблица Б, наречено Чужд ключ, което отговаря на Първичния ключ на таблица А;
- **N:N (много-към-много)** - на едно поле от таблица А отговаря едно или няколко полета от таблица Б и обратното. Подобна връзка се постига чрез междинна (свързваща) таблица.

По този начин, ако потребителят е запознат със връзките в базата данни той е в състояние да достигне до всички данни. Достигането до определени данни се извършва като се зададе кое поле, от коя таблица (таблици) ни е необходимо. Основният начин за получаване на данните е използването на стандартния език SQL (Structured Query Language). SQL е език, чрез който могат да бъдат създавани, модифицирани и поддържани релационни бази данни, както и да се извършват запитвания към тях.

Релационният модел в момента е най-често срещаният тип база от данни. Това са базите от данни, които използват свободно езика за структурирани заявки SQL. Някои много популярни търговски бази от данни, например FileMaker или Microsoft Access, не са проектирани да бъдат прилагани в Web-сайт. Те са проектирани основно за лекота на употребата, отколкото за скорост.

Обектно-ориентираните и обектно-релационните бази данни са нови и все още са в процес на разработка модели за достъп до данни. Обектно-ориентираната база от данни е предназначена да работи по-плавно с обектно-ориентирани езици за програмиране. Обектно-релационната база е хибрид, който се използва за типове данни, които не се обслужват добре от обикновените релационни бази от данни.

MySQL е проста, не усложнена база от данни за случаи, когато няма нужда от сложни търговски решения, а от добра производителност. Тя представя добри възможности за търсене в бази данни и за обслужване на Web-заявки, тъй като е бърза и надеждна. Освен това MySQL се доставя в реализации както за UNIX, така и за Windows. Съществува обединение между PHP и MySQL и платформата Apache.

Основата на всяка Web базирана информационна система е Web сървър. Web сървър е програма, която се изпълнява на даден компютър (сървър). Основното предназначение на сървъра е да приема заявки и в зависимост от типа на заявката се визуализира определена web страница или се изпълнява програма. И в двата случая сървър връща резултат. Възможно е резултата да е съобщение за грешка.

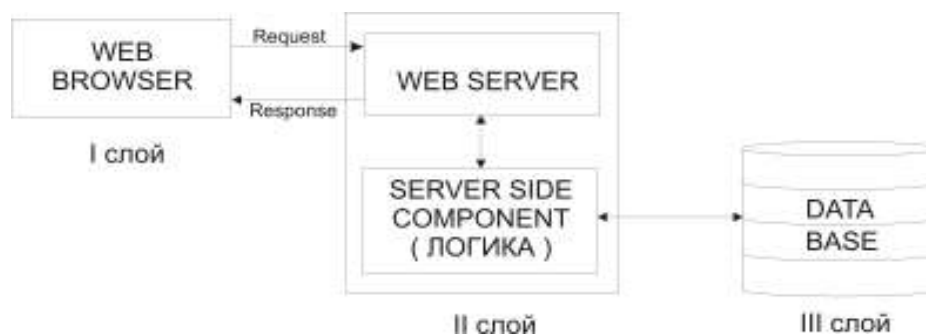
Говорейки за web базирани системи е уместно най-напред да представим концепцията за многослойна архитектура. Типичните клиент – сървър системи се причисляват към двуслойните архитектури. Приложението се изпълнява на клиентския компютър, докато базата данни е стартирана на друг сървър. Това позволява поделянето на общи данни между клиентите, но има и много недостатъци. Към тях се отнасят:

- увеличения мрежов трафик
- изисквания към клиентските компютри, свързани с техните параметри
- трудна поддръжка

За преодоляване на тези недостатъци са разработени многослойни архитектури, които най-често са трислойни. Приложението се разделя на три отделни логически нива, всяко от които е добре обособено. Първото ниво в повечето случаи представлява графичен потребителски интерфейс (GUI), които ще наричаме Web-клиент (например Web-браузър). Средното ниво е логическо и обикновено е Web-сървър, CGI-скриптове и приложни програмни интерфейси за връзка с базата данни (например Apache с модул PHP, поддържащ MySQL база данни и PHP скриптове).

Чрез заявка от потребителския интерфейс се задейства функция в логическото ниво, което извлича желаните данни, обработва ги и ги визуализира. Диференцирането на логическото ниво от потребителския интерфейс, дава по-голяма гъвкавост на интерфейса и функционалността на системата. Множество потребителски интерфейси могат да бъдат изградени и използвани без да се налага промяна в логическото ниво. Третото ниво съдържа данните на системата. Тази информация може да бъде съхранена в дадена релационна база данни от типа на Oracle, MySQL – Server и др.

Предимствата при използването на трислойна архитектура са: увеличаване на производителността и редуциране на мрежовия трафик.



Фиг. 6. Трислойна система на взаимодействие

За да преинемем от трислойна архитектура към многослойна, трябва да разширим средното ниво (логическото). В следствие на това разполагаме с няколко на брой приложни обекта, които комуникират помежду си, използвайки собствени интерфейси.

4.2.3. Дефиниране на таблиците за БД.

Форумът има следната съвкупност от обекти и за всеки един от тези обекти ще се създаде таблица в БД - Потребител, Форум, Тема, Отговор, Осведомяване по e-mail и Забраняване на IP адрес.

Таблицата USERS съдържа описание на обекта потребител (Фиг. 7)

Web базирано представяне на традициите и обичаите на България

Таблица USERS

Име на колона	Тип	Разрешава ли null	Стойност по подразбиране	Ключ	Допълнение
user_id	int(10)	Not Null		Pr.Key	auto_increment
username	varchar(40)	Not Null			
user_regdate	datetime	Not Null	0000-00-00 00:00:00		
user_password	varchar(32)	Not Null			
user_email	varchar(50)	Not Null			
user_icq	varchar(15)	Not Null			
user_website	varchar(100)	Not Null			
user_occ	varchar(100)	Not Null			
user_from	varchar(100)	Not Null			
user_interest	varchar(150)	Not Null			
user_viewemail	tinyint(1)	Not Null	0		
user_sorttopics	tinyint(1)	Not Null	1		
user_newpwdkey	varchar(32)	Not Null			
user_newpasswd	varchar(32)	Not Null			

Фиг.7.Таблица описваща обекта Потребител.

Атрибутите на таблицата са:

1. **user_id** – уникален идентификационен номер на таблицата (първичен ключ).
2. **username** – име на потребителя.
3. **user_regdate** – дата и час на регистрация на потребителя.
4. **user_password** – парола на потребителя.
5. **user_email** – e-mail на потребителя.
6. **user_icq** – ICQ номер на потребителя.
7. **user_website** – Web сайт на потребителя.
8. **user_occ** – професия на потребителя.
9. **user_from** – от коя страна е по народност потребителя.
10. **user_interest** – интереси на потребителя.
11. **user_viewemail** – дали e-mail на потребителя ще е видим за останалите потребители.
12. **user_sorttopics** – как потребителя желае да вижда статиите – подредени по нови теми или нови отговори.
13. **user_newpwdkey** – при загуба на парола, стойността на новата парола, която автоматично се генерира.
14. **user_newpasswd** – ако потребителят смени автоматично генерираната парола пази стойността на новата

Web базирано представяне на традициите и обичаите на България

Един потребител може да създава много теми и да отговаря на тях. Таблицата е създадена чрез следната заявка:

```
CREATE TABLE users (  
  user_id int(10) NOT NULL auto_increment,  
  username varchar(40) NOT NULL default "",  
  user_regdate datetime NOT NULL default '0000-00-00 00:00:00',  
  user_password varchar(32) NOT NULL default "",  
  user_email varchar(50) NOT NULL default "",  
  user_icq varchar(15) NOT NULL default "",  
  user_website varchar(100) NOT NULL default "",  
  user_occ varchar(100) NOT NULL default "",  
  user_from varchar(100) NOT NULL default "",  
  user_interest varchar(150) NOT NULL default "",  
  user_viewemail tinyint(1) NOT NULL default '0',  
  user_sorttopics tinyint(1) NOT NULL default '1',  
  user_newpwdkey varchar(32) NOT NULL default "",  
  user_newpasswd varchar(32) NOT NULL default "",  
  PRIMARY KEY (user_id)  
) TYPE=MyISAM;
```

Таблицата FORUMS съдържа описание на обекта форум (Фиг.8)

Таблица FORUMS

Име на колона	Тип	Разрешава ли null	Стойност по подразбиране	Ключ	Допълнение
forum_id	int(10)	Not Null		Pr.Key	auto_increment
forum_name	varchar(150)	Not Null			
forum_desc	text	Not Null			
forum_order	int(10)	Not Null	0		
forum_icon	varchar(255)	Not Null	default gif		

Фиг.8. Таблица описваща обекта Форум.

Атрибутите на таблицата са:

Web базирано представяне на традициите и обичаите на България

1. **forum_id** – уникален идентификационен номер на таблицата (първичен ключ).
2. **forum_name** – име на форума.
3. **forum_desc** – описание на форума.
4. **forum_order** – ред на форума.
5. **forum_icon** – икона на форума.

Един форум може да бъде създаден от потребител с администраторски права. Таблицата е създадена чрез следната заявка:

```
CREATE TABLE forums (  
forum_id int(10) NOT NULL auto_increment,  
forum_name varchar(150) NOT NULL default "",  
forum_desc text NOT NULL,  
forum_order int(10) NOT NULL default '0',  
forum_icon varchar(255) NOT NULL default 'default.gif',  
PRIMARY KEY (forum_id)  
) TYPE=MyISAM;
```

Таблицата TOPICS съдържа описание на обекта тема (Фиг. 9)

Таблица TOPICS

Име на колона	Тип	Разрешава ли null	Стойност по подразбиране	Ключ	Допълнение
topic_id	int(10)	Not Null		Pr.Key	auto_increment
topic_title	varchar(100)	Not Null			
topic_poster	int(10)	Not Null	0		
topic_poster_name	varchar(40)	Not Null	Anonymous		
topic_time	datetime	Not Null	0000-00-00 00:00:00		
topic_views	int(10)	Not Null	0		
forum_id	int(10)	Not Null	1	Foreign Key	
topic_status	tinyint(1)	Not Null	0		
topic_last_post_id	int(10)	Not Null	1		

Фиг.9.Таблица описваща обекта Тема .

Атрибутите на таблицата са:

Web базирано представяне на традициите и обичаите на България

1. **topic_id** – уникален идентификационен номер на таблицата (първичен ключ).
2. **topic_title** – заглавие на темата.
3. **topic_poster** – идентификационен номер на потребителя създал темата.
4. **topic_poster_name** – име на потребителя създал темата
5. **topic_time** – дата и час на създаване на темата.
6. **topic_views** – брой на посещения на темата
7. **forum_id** - идентификационен номер на форума, към който принадлежи темата (чужд ключ).
8. **topic_status** – статус на темата (деликатна или не е тя).
9. **topic_last_post_id** – идентификационен номер на последния отговор по темата.

Една тема може да бъде създадена от един потребител и може да притежава много отговори. IP адресът, от който е създадена темата може да е само един. Таблицата е създадена чрез следната заявка:

```
CREATE TABLE topics (  
  topic_id int(10) NOT NULL auto_increment,  
  topic_title varchar(100) NOT NULL default "",  
  topic_poster int(10) NOT NULL default '0',  
  topic_poster_name varchar(40) NOT NULL default 'Anonymous',  
  topic_time datetime NOT NULL default '0000-00-00 00:00:00',  
  topic_views int(10) NOT NULL default '0',  
  forum_id int(10) NOT NULL default '1',  
  topic_status tinyint(1) NOT NULL default '0',  
  topic_last_post_id int(10) NOT NULL default '1',  
  PRIMARY KEY (topic_id),  
  KEY topic_id(topic_id),  
  KEY forum_id(forum_id)  
) TYPE=MyISAM;
```

Таблицата POSTS съдържа описание на обекта отговор (Фиг. 10)

Таблица POSTS

Име на колона	Тип	Разрешава ли null	Стойност по подразбиране	Ключ	Допълнение
post_id	int(10)	Not Null		Pr.Key	auto_increment
forum_id	int(10)	Not Null	1	Foreign Key	
topic_id	int(10)	Not Null	1	Foreign Key	
poster_id	int(10)	Not Null	0	Foreign Key	
poster_name	varchar(40)	Not Null	Anonymous		
post_text	text	Not Null			
post_time	datetime	Not Null	0000-00-00 00:00:00		
poster_ip	varchar(15)	Not Null			
post_status	tinyint(1)	Not Null	0		

Фиг.10.Таблица описваща обекта Отговор .

Атрибутите на таблицата са:

1. **post_id** – уникален идентификационен номер на таблицата (първичен ключ).
2. **forum_id** – идентификационен номер на форума, към който принадлежи отговора (чужд ключ).
3. **topic_id** – идентификационен номер на темата, към която принадлежи отговора (чужд ключ).
4. **poster_id** – идентификационен номер на потребителя, който е създал отговора (чужд ключ).
5. **poster_name** – името на потребителя, който е създал отговора.
6. **poster_text** – съдържание на отговора.
7. **poster_time** – час и дата на създаване на отговора.
8. **poster_ip** – IP адрес на потребителя създал отговора.
9. **post_status** – статус на отговора

Един отговор може да бъде пуснат от един потребител и да принадлежи на един форум и съответна тема принадлежаща към дадения форум. Един потребител може да отговори на много теми. Компютърът, от който е изпратен отговора може

Web базирано представяне на традициите и обичаите на България

да е само един и съответно и IP адреса му. Таблицата е създадена чрез следната заявка:

```
CREATE TABLE posts (  
  post_id int(11) NOT NULL auto_increment,  
  forum_id int(10) NOT NULL default '1',  
  topic_id int(10) NOT NULL default '1',  
  poster_id int(10) NOT NULL default '0',  
  poster_name varchar(40) NOT NULL default 'Anonymous',  
  post_text text NOT NULL,  
  post_time datetime NOT NULL default '0000-00-00 00:00:00',  
  poster_ip varchar(15) NOT NULL default '',  
  post_status tinyint(1) NOT NULL default '0',  
  PRIMARY KEY (post_id),  
  KEY post_id(post_id),  
  KEY forum_id(forum_id),  
  KEY topic_id(topic_id),  
  KEY poster_id(poster_id)  
) TYPE=MyISAM;
```

Таблицата SEND_MAILS съдържа описание на обекта Осведомяване по e-mail (Фиг. 11)

Таблица SEND_MAILS

Име на колона	Тип	Разрешава ли null	Стойност по подразбиране	Ключ	Допълнение
id	int(11)	Not Null		Pr.Key	auto_increment
user_id	int(10)	Not Null	1		
topic_id	int(10)	Not Null	1		

Фиг.11.Таблица описваща обекта Осведомяване по e-mail .

Атрибутите на таблицата са:

1. **id** – уникален идентификационен номер на таблицата (първичен ключ).

2. **user_id** – идентификационен номер на потребителя, който желае услугата осведомяване по e-mail.
3. **topic_id** – идентификационен номер на темата, към която даденият потребител се е абонира да получава информационни писма.

Услугата осведомяване по e-mail е валидна само за един потребител и при обновяване на една тема. Ако даден потребител желае това за няколко теми трябва да се абонира за тази услуга по отделно. Таблицата е създадена чрез следната заявка:

```
CREATE TABLE send_mails (  
  id int(11) NOT NULL auto_increment,  
  user_id int(10) NOT NULL default '1',  
  topic_id int(10) NOT NULL default '1',  
  PRIMARY KEY (id)  
) TYPE=MyISAM;
```

Таблицата BANNED_IP съдържа описание на обекта Забрана на IP адрес (Фиг. 12)

Таблица BANNED_IP

Име на колона	Тип	Разрешава ли null	Стойност по подразбиране	Ключ	Допълнение
id	int(10)	Not Null		Pr.Key	auto_increment
banip	varchar(15)	Not Null			

Фиг.12. Таблица описваща обекта Забрана на IP адрес.

Атрибутите на таблицата са:

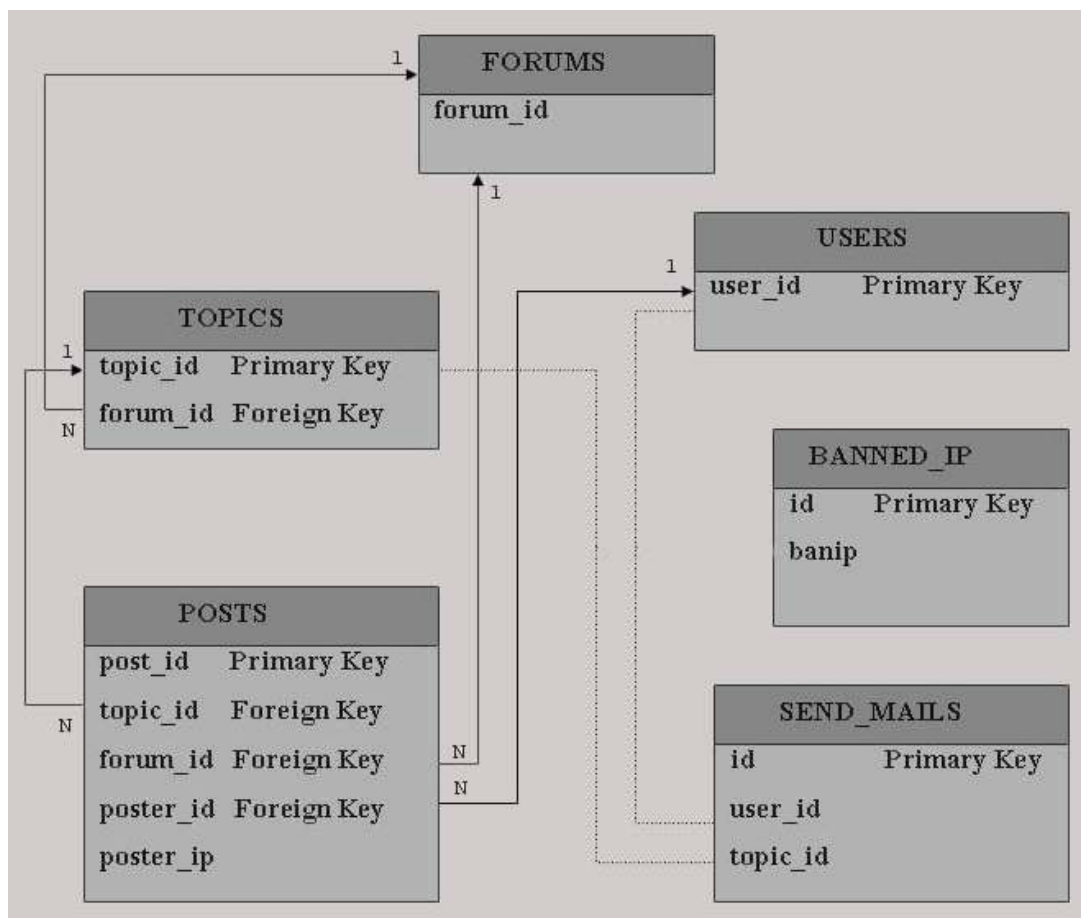
1. **id** – уникален идентификационен номер на таблицата (първичен ключ).
2. **banip** – забраненият IP номер

Web базирано представяне на традициите и обичаите на България

По преценка на администратора на форума, създаването на теми или отговори от даден IP адрес може да се забрани. Таблицата е създадена чрез следната заявка:

```
CREATE TABLE banned (  
  id int(10) NOT NULL auto_increment,  
  banip varchar(15) NOT NULL default "",  
  PRIMARY KEY (id)  
) TYPE=MyISAM;
```

Връзките между обектите от форума са едно към едно и едно към много.. Релациите между отделните обекти са показани на фиг. 13.



Фиг.13.Връзки между таблиците.

Всяка една от таблиците е в трите нормални форми:

- Първа нормална форма - Казваме, че една таблица се намира в Първа нормална форма, ако всеки един от атрибутите ѝ има единствена стойност за всеки екземпляр на таблицата.
- Втора нормална форма - Казваме, че една таблица се намира във Втора нормална форма, ако всеки един от атрибутите ѝ зависи от целия първичен ключ на таблицата.
- Трета нормална форма - Казваме, че една таблица се намира в Трета нормална форма, ако нито един атрибут, принадлежащ на първичният ѝ ключ, не зависи от друг атрибут, принадлежащ на нейния първичен ключ.

4.3. Проектиране на Flash клипа.

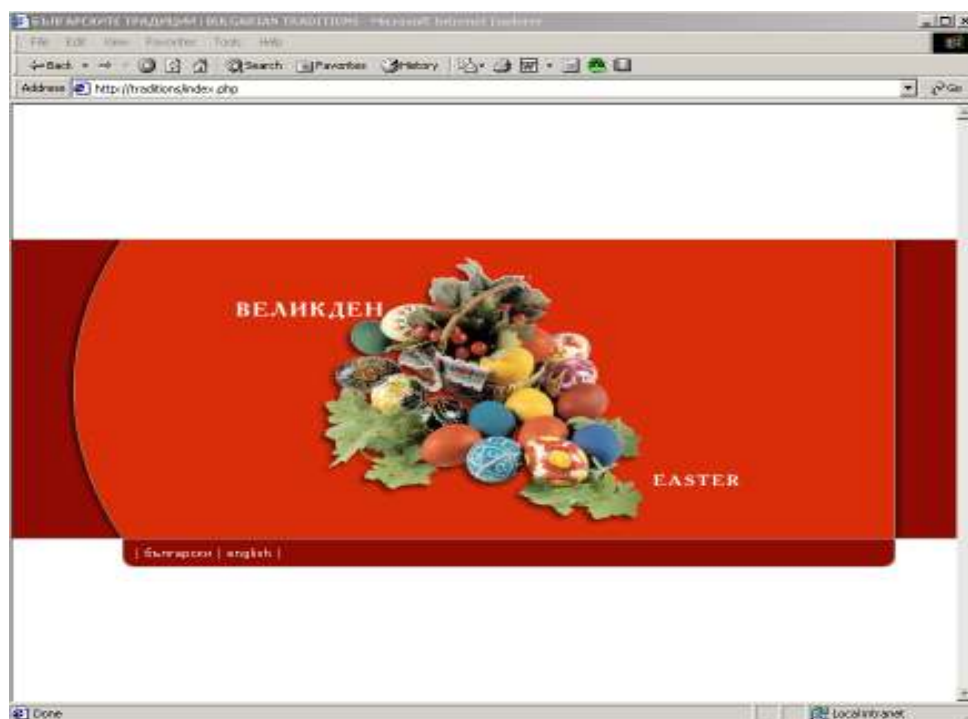
Създаването на уводна анимация към web сайта за българските традиции и обичаи възникна от необходимостта да се задържи вниманието на потребителя още при влизането в страниците му. За целта се постаряхме тя да отговаря на следните критерии:

- Трябва да е с кратка продължителност. Ако анимацията е по – дълга като времетраене има риск потребителя да бъде отегчен. Също така това може да доведе до по – голям размер на SWF файла, а оттам и до по-дълга фаза на зареждане, което би затруднило потребителите с влошено качество на Internet връзката и би ги отказало от разглеждането на сайта.
- Трябва да е интерактивна и забавна и да подскаже на потребителя на каква тема е сайта. За това в уводната анимация са вмъкнати снимкови материали от най – известните обичай на българите като кукери, кръщение и Великден. Появяването на всяко графично изображение е придружено със текст, обясняващ какъв празник описва то.
- Тъй като сайта е ориентиран да предложи информация не само на българските потребители, уводната страница трябва ясно да

показва, че информацията е налична освен на български и на английски език.

- Трябва да има възможност по всяко време потребителя да се откаже от разглеждането на уводната анимация и да премине по нататък по страниците на сайта. Той има възможност да направи това посредством хипервръзките “български” и “english”, които го отвеждат съответно в българската и английската версия.
- Цветовата гама на анимацията трябва да е в унисон с интерфейса на останалата част от сайта.

На фигура 14 може да се види кадър от уводната анимация.



Фиг. 14. Кадър от уводната анимация.

5. Реализация на форума.

Описанието на приложението форум е разделено на три части. В първата част е описан графичния интерфейс и функционалността на форума, във втората част е описано как е реализирана функционалността и как е структуриран кодът на приложението, а третата част представлява ръководство на потребителя за

първоначални настройки на приложението Третата част може да откриете в Приложение 1.

5.1.Преглед на интерфейса и функционалността.

5.1.1.Страници достъпни за обикновения потребител.

Графичния интерфейс се състои от няколко страници.Страниците могат да бъдат разделени на такива, които са видими за обикновения потребител и такива – видими за администратора. В тази точка ще опишем всяка една от тях.Заглавната част на всички страници е с емблемата на сайта за българските традиции.и във всеки един момент потребителя може да отиде в английската версия на форума.

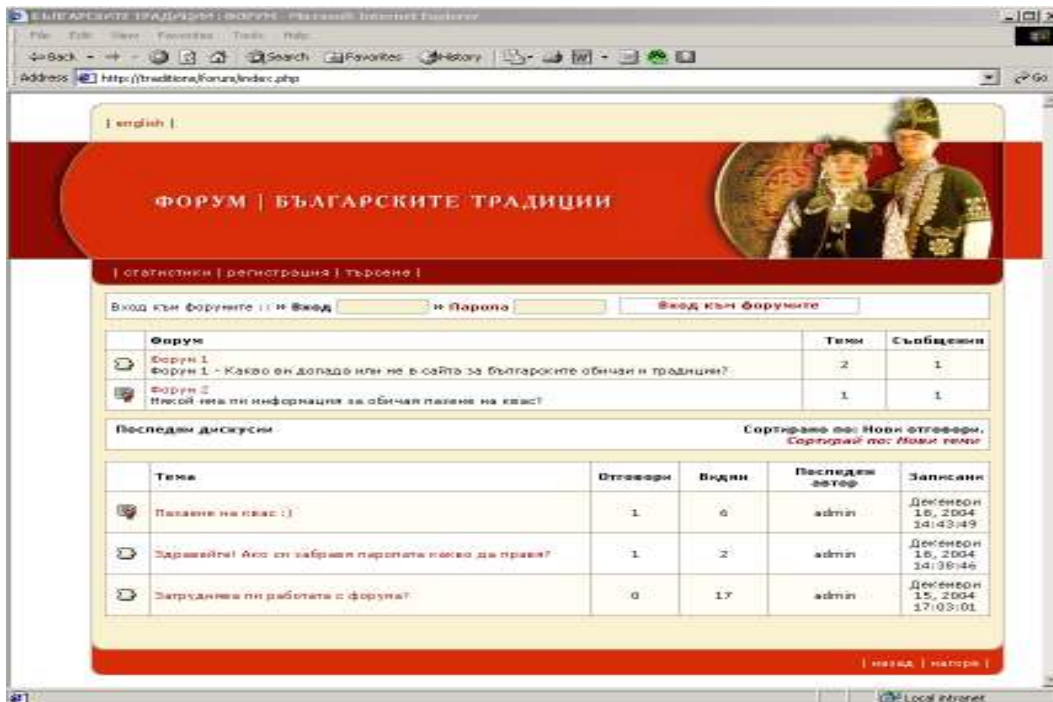
- **Начална страница**

Първата страница, която виждат потребителите, е страница със списък на форумите и темите принадлежащи към тях, както и тяхното описание (фиг.15). Принадлежността на даден отговор или тема към даден форум е реализирана чрез емблема, която е показана пред името на съответния обект. Информацията предоставена за форумите съответно е брой принадлежащи към него теми и отговори, а за темите – брой принадлежащи отговори, брой на посещенията на темата, име на автора и дата и час на публикация. Посетителяш на форума ще има възможност да види списъка от теми сортиран по актуалност на отговори и по актуалност на теми. От тази страница посетителите имат възможност да влязат в даден форум или тема, като щракнат върху съответното име.

Също така ако посетителя е вече регистриран в страниците на форума има възможност да влезе неговите страници със своето потребителско име и парола. Ако не е регистриран може да направи това използвайки хипервръзката Регистрация.

Потребителят има възможност да търси дадена информация във форума или да види статистиките за най – посещаваните теми съответно използвайки препратките Статистики и Търсене.

Връзките Регистрация, Статистики и Търсене се намират в менюто на форума, което в зависимост коя страница е показана съдържа различна информация.

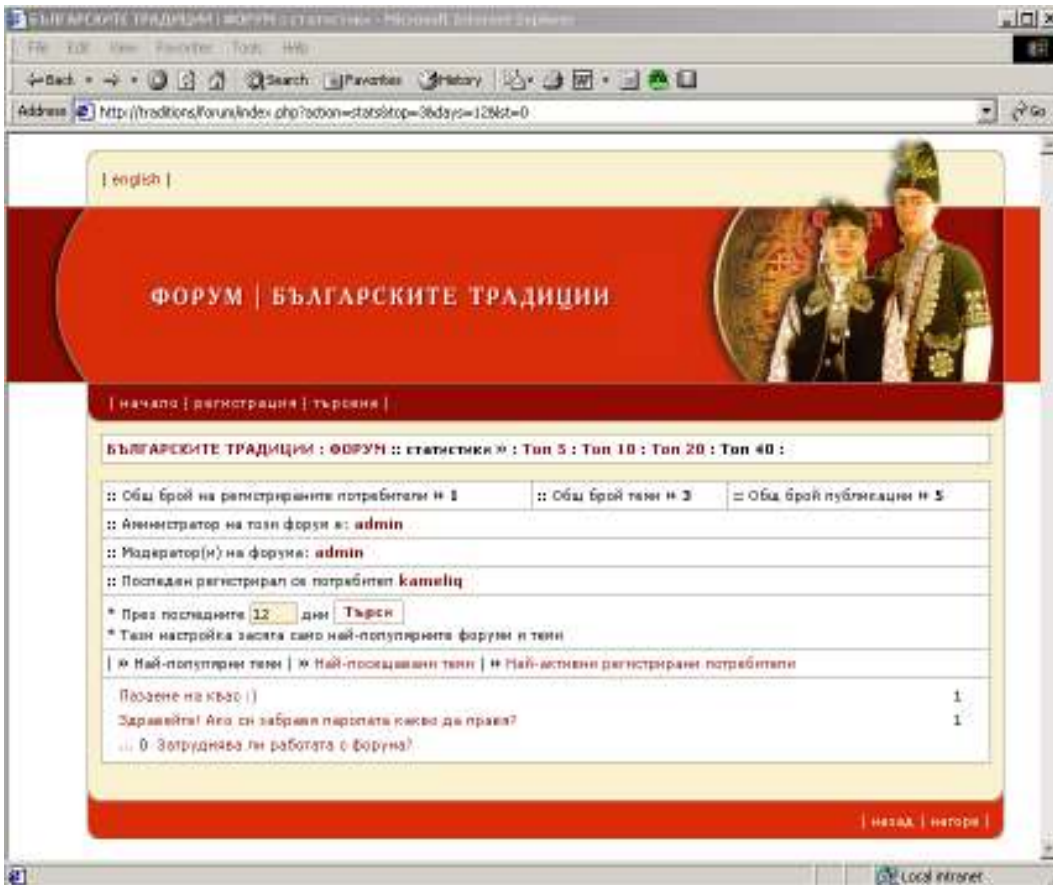


Фиг.15. Начална страница на форума.

- Страница Статистики

Страницата статистики показва най – актуалната информация за форумите (фиг.16), а това е – брой на регистрираните до момента потребители; името на администратора на форума; името на последния регистриран потребител; общ брой теми; общ брой отговори; най – активни регистрирани потребители (активността се изчислява в зависимост от броя публикувани теми и отговори); най – популярни теми (популярността се изчислява в зависимост колко пъти е видяна темата); последните пет, десет, двадесет или четиридесет теми.

Потребителят има възможност да търси най – популярните теми за даден период от време. Примерно 12 дни както е показано на фиг.16.



Фиг.16.Страница Статистики на форума.

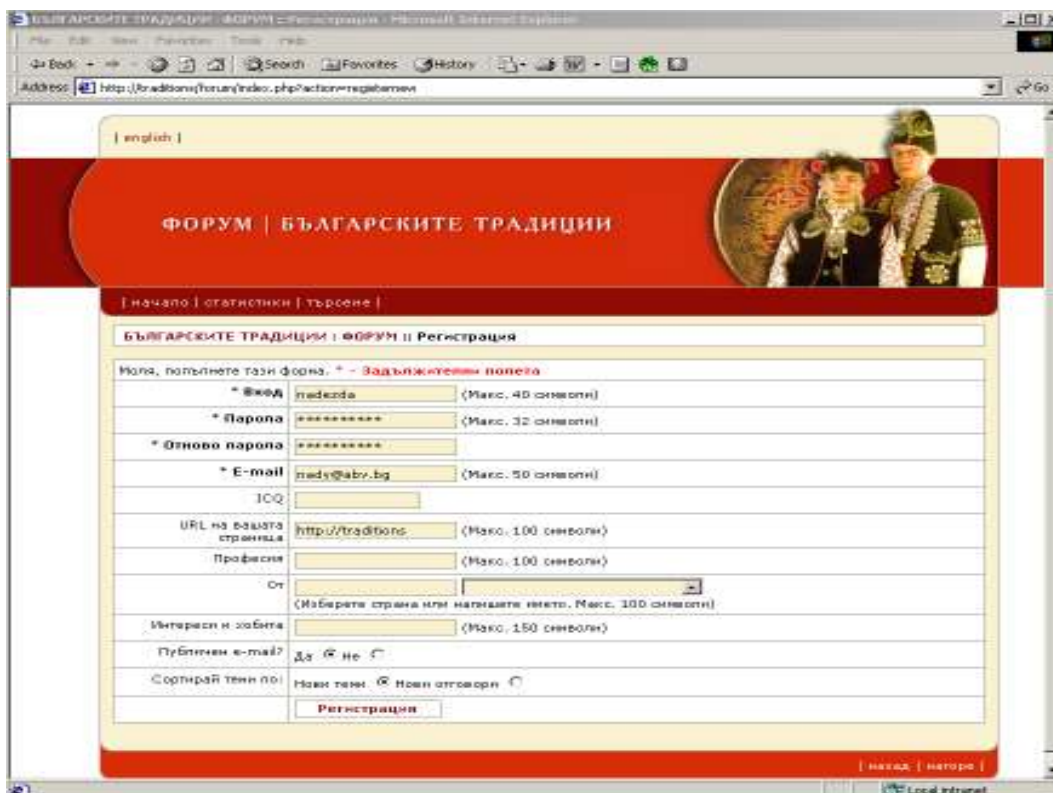
• Страница Регистрация

Посредством тази страница даден посетител може да се регистрира като потребител на форума. За целта трябва да предостави определена информация за себе си като попълни формата показана на фиг.16. Формата има задължителни полета за попълване (потребителско име, парола, е – mail адрес)и незадължителни (ICQ номер, интернет адрес на страница, професия, родна страна, интереси и хобита, дали желае е – mail адресът му да е видим за останалите потребители и как да се сортиран списъка от темите).

При неправилно въведена информация, излиза съобщение което указва къде е грешката. Възможни грешки са: непозволени символи за потребителско име и парола, потребителското име вече заето, неправилен е – mail адрес, първоначално и повторно въведените пароли не съвпадат. При регистрация дадения потребител

Web базирано представяне на традициите и обичаите на България

получава осведомително писмо, което дава информация за потребителското име и парола, на указания от него е – mail адрес при регистрацията.



The screenshot shows a web browser window with the address bar displaying "http://traditions/forum/index.php?action=registernew". The page title is "БЪЛГАРСКИТЕ ТРАДИЦИИ | ФОРУМ". The main content area is a registration form with the following fields and options:

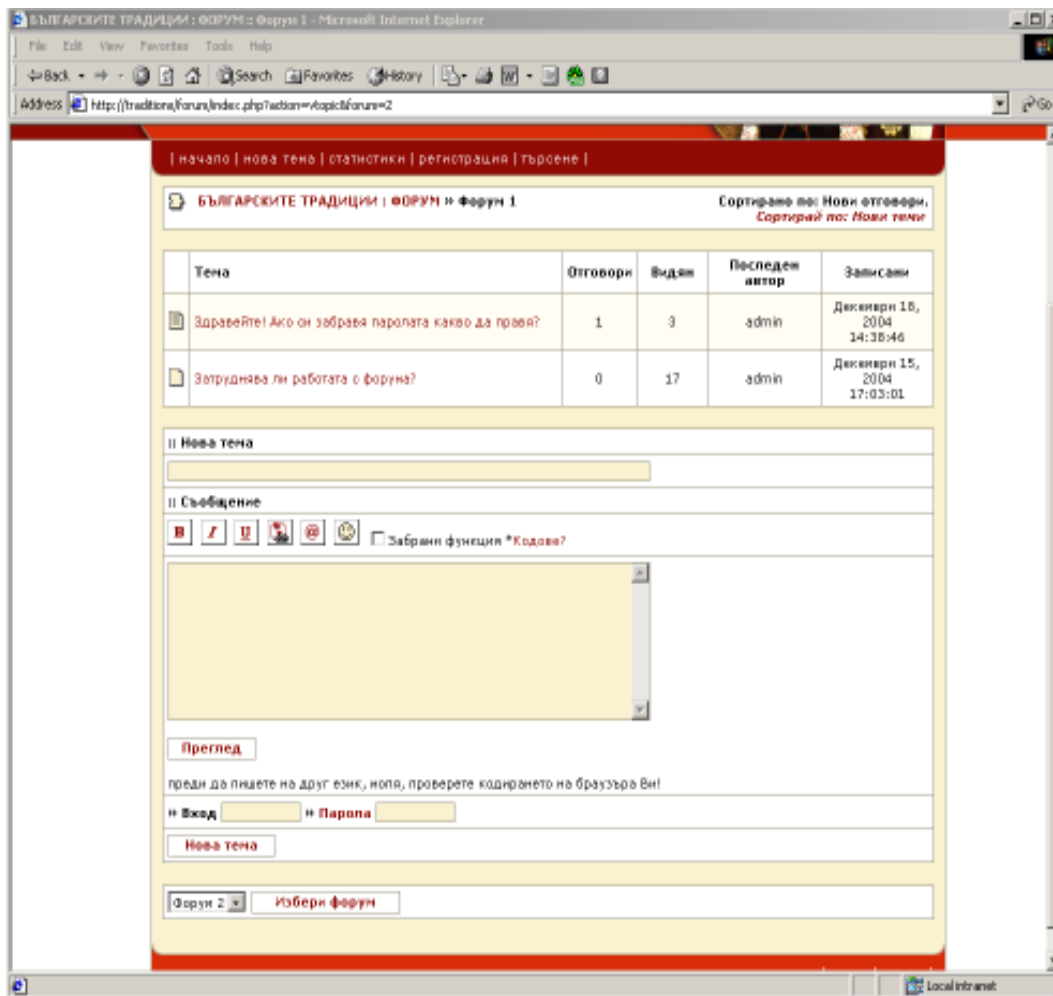
- Имя: (Макс. 40 символа)
- * Парола: (Макс. 32 символа)
- * Отново парола:
- * E-mail: (Макс. 50 символа)
- ICQ:
- URL на вазата страницата: (Макс. 100 символа)
- Професия: (Макс. 100 символа)
- От: (Изберете страна или напишете името, Макс. 100 символа)
- Интереси и хобита: (Макс. 150 символа)
- Публичен e-mail? Да Не
- Сортирай теми по: Нови теми Нови отговори

A "Регистрация" button is located at the bottom of the form.

Фиг.17.Страница Регистрация на потребител.

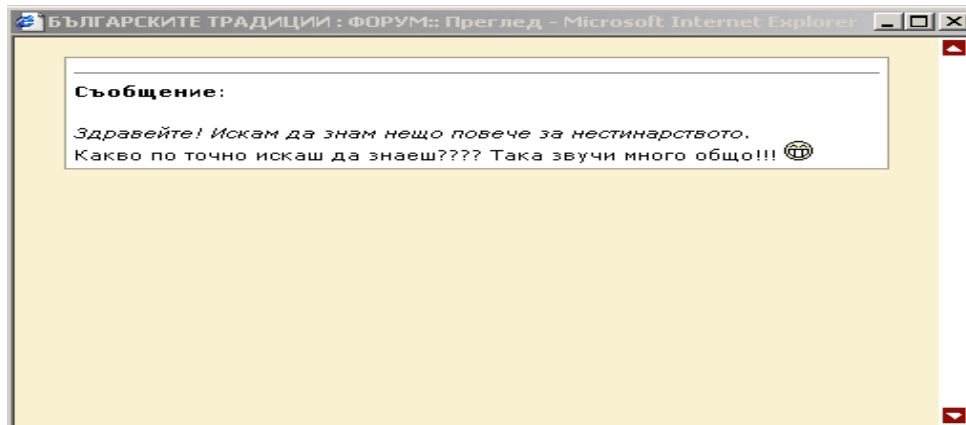
- **Страница за детайлите на форум.**

Тази страница показва списък от всички теми и отговори принадлежащи на дадения форум, както и дава възможност за добавяне на нова тема. Необходимата информация, която потребителя трябва да въведе е заглавие и съдържание на темата. Също така потребителя може да направи съдържанието на темата да изглежда по – добре, използвайки кодове заместващи HTML етикети. По – подробна информация за наличните кодове може да се достигне чрез хипервръзката Кодове, а те са – удебеляване, накланяне и подчертаване на текст. Посетителят има възможност да вмъкне още Web и e – mail адрес, и подсили своето мнение с картина изразяваща дадена емоция . Ако потребителят не желае да форматира текста на съобщението, тогава може да забрани използването на кодовете.



Фиг.18. Страница Детайли на форум.

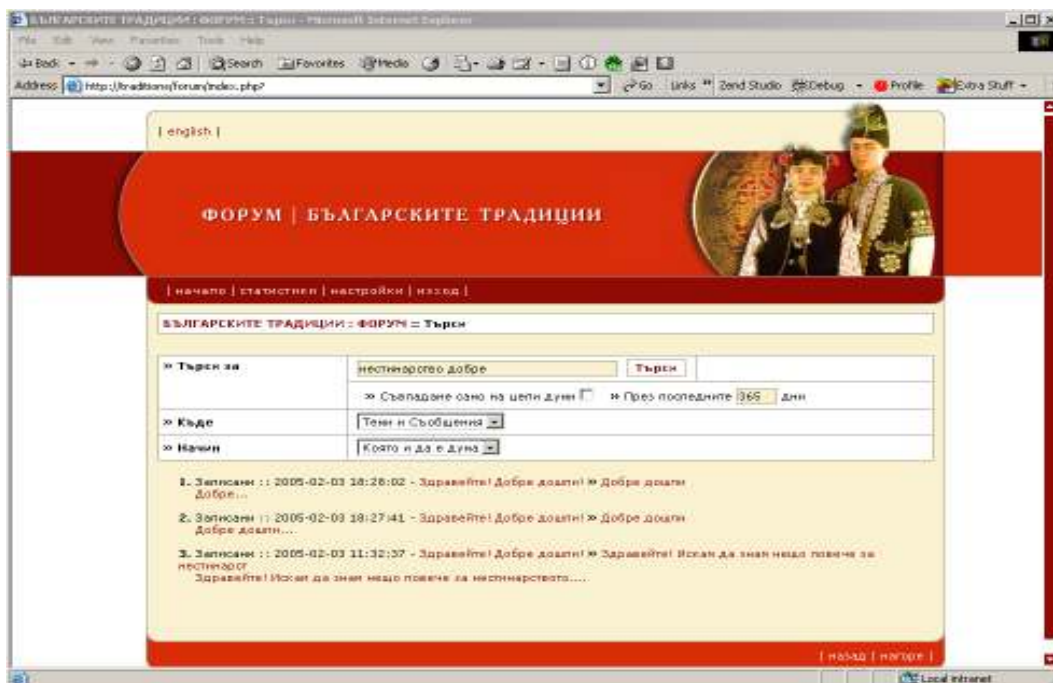
При добавяне на нова темата, въвеждането на заглавието е задължително и то ще стане име на темата. Ако потребителя отговаря на вече съществуваща тема не е нужно да въвежда заглавие. Когато потребителя не е анонимен може да включи или изключи услугата e-mail осведомяване, чрез която всеки път когато темата е модифицирана ще получава писмо уведомяващо го за дадената модификация. При отговор на дадена тема може да се цитира част от нея в отговора чрез линката “Цитат” намираща се на горния ляв ъгъл на всяко съобщение. Даден потребител може да редактира и да затвори своя тема или отговор. За целта трябва да е влязъл в страниците на форума със своето име и парола. Също така дадената тема или отговор може да бъде предварително видяна с помощта на бутона “Преглед” (фигура. 19).



Фиг.19. Страница Преглед на съобщение

- Страница Търсене.

Всеки потребител има възможност да търси в страниците на форума определена информация. Търсенето може да се направи по автор, по дума или фраза в текста на съобщението, по заглавие, по съвпадение на цяла дума или на част от нея, по времето в което са въведени (фигура 20).

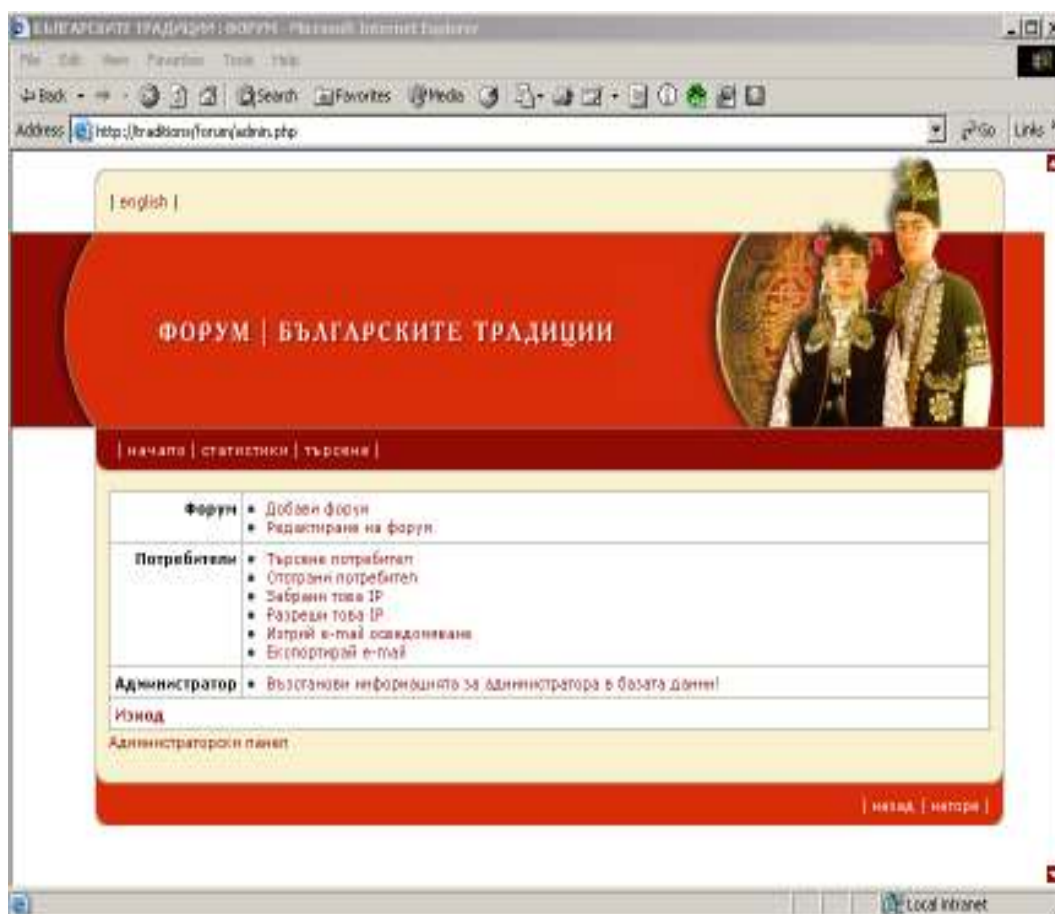


Фиг.20. Страница Търсене.

Менюто Настройки е видимо, когато даден потребител е влязъл в страниците на форума със своето име и парола, и показва личните му данни. Всички те могат да бъдат обновени с изключение на потребителското име. Ако потребителя желае да напусне страниците на форума може да направи това чрез менюто Изход.

5.1.2. Страници достъпни за администратора.

Всички страници достъпни за обикновения потребител са такива и за администратора. Когато даден посетител влезе в страниците на форума като администратор в долния край на всяка страница е достъпна препратката Администраторски панел (фигура 21). Страницата Администраторски панел дава достъп до разширените администраторски функции.



Фиг.21. Страница Администраторски панел.

Администраторът има следните възможности: да търси даден потребител по неговия е – mail и потребителско име, да изтрие даден потребител, да забрани или разреши добавянето на теми от даден IP адрес, да спре e-mail осведомяването до даден адрес.

- **Страница Добавяне на форум.**

От този екран може да се добави нов форум като задължителната информация, която трябва да се въведе е заглавието и описанието му. Потребителят има възможност да избере и икона на дадения форум.



Фиг.22. Страница Добавяне на форум.

- **Страница Редактиране на форум.**

Даден съществуващ форум може да бъде редактиран - могат да бъдат променени неговото заглавие, описание, икона и подредба. При изтриване на даден форум автоматично се изтриват всички принадлежащи му теми и отговори.

- **Страница Редактиране на тема и отговор.**

Дадена тема може да бъде затворена, изтрита, и преместена към друг форум, ако такъв съществува. Тези функции са достъпни чрез съответните препратки намиращи се в долния край на страницата Детайли на форум.

5.2.Реализация на функционалността.

Тук няма да се спрем подробно на описанието на кода на всеки PHP файл, но ще опишем как са реализирани някои от основните функции във форума.

Приложението форум има много повече файлове от тези, които бяха описани в точка 5.1 от настоящата дипломна работа. При писането на кода се възползвахме от възможността на PHP за повторно използване на код. PHP улеснява използването на части от код или HTML в един документ с помощта на конструкциите `include` и `require`. Разликата между тях е особено важна. Конструкцията `include` е условна – файлът се чете при всяко срещане на `include`. Т.е ако конструкцията не бъде достигната, файлът няма да бъде включен. Но ако един и същ файл е включен два пъти с конструкцията `include`, PHP ще го прочете два пъти. Включените с помощта на `require` файлове се прочитат само веднъж. Ние използвахме и двете конструкции в зависимост от случая. Много по – добре е кодът да се раздели на функции и свързаните елементи да се поставят в `include` файлове, от колкото цялата функционалност да се съхрани в един огромен файл. Например всички функции, които са свързани с база данни, сме поставили във файл с име “`setup_mysql.php`”. По този начин кода става по лесен за четене, разбиране и повторно използване. Например “`setup_mysql.php`” се използва във всеки скрипт, в който трябва да се свържем към база от данни на форума. Също така по този начин кодът става по – лесен за поддръжка. Ако е необходимо да се промени начина, по който правим обръщение към нашата база от данни, ще го променим само на едно място. Всеки един такъв включен файл със функции ще наричаме библиотека. Файловете, които реално извършват дадено действие

(примерно добавят нова тема към даден форум) и извикват библиотеките посредством include конструкцията ще наричаме приложения.

- **Имплементиране на идентификацията на потребителите във форума и управление на сесията на даден потребител.**

Тъй като искаме да обвържем всеки потребител с персонализираща информация, ще съхраняваме потребителското име и парола в MySQL база данни в таблицата users, която бе описана в точка 4.2.3. Чрез това потребителско име и парола ще идентифицираме дадения потребител. Налага се да сложим ограничения за дължината и формата на потребителското име и парола. Трябва да съхраняваме паролата в криптиран формат по причини за сигурност. Потребителите трябва да могат да влизат с името и паролата, получени в процеса на регистрация. Потребителите трябва да могат и да излизат след като завършат работата си със сайта. Това не е особено важно, когато те използват сайта от собствения си домашен компютър, но е много важно от гледна точка на сигурността, ако използват сайта от споделен компютър. Сайтът трябва да може да проверява дали даден потребител е влязъл или не, и да осъществява достъп до данните му. Потребителите трябва да могат да сменят своята парола за по – голяма сигурност. Понякога е възможно потребителите да забравят своите пароли. Те трябва да могат да ги възстановяват без необходимостта от личното съдействие на администратора. Това става чрез изпращане на парола на e-mail адреса на потребителя зададен при регистрацията му. Тъй като паролите се съхраняват в криптиран формат и не могат да бъдат декодирани е необходимо да се генерира нова парола.

Сесиите в PHP се управляват чрез уникален идентификационен номер (ID), представляващ криптографско случайно число. Този сесийен ID се генерира от PHP се съхранява при клиента за времето на живот на сесията. Той може да бъде съхранен в бисквитка на потребителския компютър или предаван чрез URL адреси. Сесийният ID действа като ключ и позволява да се регистрират определени променливи като така наречените сесийни променливи. Тяхното съдържание се съхранява на сървъра. Единствената достъпна информация за клиента е сесийният ID. Ако по време на връзка със сайта сесийният ID е достъпен чрез бисквитка или URL адрес, може да се установи достъп до сесийните променливи, съхранени на

сървъра за тази сесия. Бисквитката представлява данни, които скриптовете могат да съхранят на клиентската машина. Ръчното установяване на бисквитки става чрез функцията `setcookie()`. Управлението на сесии се използва най – често за следене на потребителите след тяхната автентикация. Ние сме реализирали това чрез съчетанието от MySQL и употребата на сесии.

Главният PHP файл, който обработва всички действия желани от обикновения потребител е `“index.php”`. В началото на `“index.php”` чрез функцията `include` са включени следните библиотеки:

- `“setup_mysql.php”` – съдържа всички функции за работа с базата данни като свързване с базата данни и всички възможни заявки, които са ни необходими за работата с базата данни на форума (чрез функцията `DB_query`)
- `“codes.php”` – съдържа две функции за кодиране и декодиране на низове, които се използват за обработка на низове преди да се съхранят в базата данни или да се извлекат от базата данни.
- `“functions.php”` – съдържа всички функции, които не могат да се причислят към определена група, като например функциите за генериране на HTML от шаблони, прашане на поща и работа с дати.
- `“specials.php”` – съдържа дефинирането на масивите, които са ни необходими.
- `“plugins.php”` – включва PHP файловете `hack_smilies.php` и `hack_preview.php`, които са ни необходими за да покажем как ще изглежда дадена тема и списъка с емоциите.
- `“$lang.php”` – дефинира езика, който сме избрали.
- `“setup_options.php”` – файла, който определя всички първоначални настройки на форума като кой е избрания език, как се казва базата данни, личните данни на администратора (име, парола и e-mail адрес), дали е позволено изпращането на поща, как се казват таблиците в базата данни и т.н. По – подробно описание на този файл може да видите в приложение 1 от настоящата дипломна работа.

Този файл (“index.php”) определя какво действие да извърши чрез две главни променливи предавани като скрити полета от HTML формите на форума – \$mode и \$action. Променливата \$mode се предава и има стойност само, когато даден потребител желае да влезе в страниците на форума със своите потребителски данни (login) или да напусне страниците на форума (logout). Променливата \$action има стойност при всички останали действия на потребителя, като например изтриване или добавяне на тема. Когато даден потребител желае да се регистрира стойността на променливата \$mode няма значение. Стойността на \$action е равна на register. Всички функции свързани със регистрацията на потребител са отделени в отделен PHP файл (“func_regusr.php”), който се извиква с конструкцията require, ако действието желано от потребителя е регистрация.

```
elseif($action=='register') {$step=1;require('./func_regusr.php');}
```

Функциите от файла “func_regusr.php” проверяват дали всички данни въведени от потребителя са коректни и дали в базата данни не съществува вече потребител със същото име. Тук се извиква и вградената хеш функция за криптиране md5, която създава за дадено съобщение (в случая паролата на потребителя) т.нар хеш стойност. Ползваме тази функция, защото за дадени входни данни изходните данни са винаги едни и същи. А това по късно ще ни е от полза (при опит на вече регистриран потребител да влезе в страниците на форума със своето име и парола) за проверката за съвпадение на въведената парола и тази съхранена в базата данни в криптиран формат. Ако такъв потребител съществува или някой от данните не са коректни се появява съобщение, което указва къде е грешката. Следният код проверява дали името, паролата, е – mail адреса, професията и web страницата са коректни.

```
<?php
function checkUserData($userData, $act) {
    global $userRegName, $disallowNames;
    $userRegExp = "#^[\".$userRegName.\"]{3,40} $#";
    if (!preg_match($userRegExp,$userData[1]) or in_array($userData[1],$disallowNames)) {
return 1; }
    elseif ($act=='reg' and !eregi("[A-Za-z0-9_]{5,32}$", $userData[2])) { return 2; }
```

```
elseif ($act=='upd' and $userData[2]!=" and !ereg("^[A-Za-z0-9_]{5,32}$", $userData[2])) {  
return 2; }  
elseif ($userData[2] != $userData[3]) { return 3; }  
elseif (!ereg("^[0-9a-z]+([._][0-9a-z]+)*_?@[0-9a-z]+([._][0-9a-z]+)*[.][0-9a-z]{2}[0-9A-  
Z]?[0-9A-Z]?$", $userData[4])) { return 4; }  
elseif ($userData[5] != " and !ereg("^[0-9]*$", $userData[5])) { return 5; }  
elseif ($userData[6] != " and !ereg("^(f|ht)tp[s]?:\V[^\>]+$", $userData[6])) { return 6; }  
else { return "ok"; }  
}  
?>
```

Функцията `checkUserData` връща резултат ОК ако всички данни са коректни в противен случай връща номер, който указва какво съобщение за грешка ще бъде генерирано. За да проверим коректността на входните данни сме се възползвали от вградената възможност на PHP да обработва регулярни изрази, които са средство за допълнителна работа с низове и проверка за съответствие. Един регулярен израз по същество представлява маска – множество от символи, описващи естеството на низа, който се търси. (Кастането, 2001, стр.246). Очевидно не можем да направим проверка дали дадения потребител е въвел коректен е – mail адрес като правим директна проверка за съвпадение на отделни думи. За да направим това трябва да проверим всички възможни адреси или имена, или професии и т.н. За целта използваме символните класове, които са удобен начин за описание на стойностите, за които се опитваме да открием съвпадение. За да създадем символен клас, който да съответства на коя да е гласна, поставяме всички гласни в квадратни скоби (Кастането, 2001, стр.247). Освен това можем да използваме и тире, за да представим поредица от символи. Примерно символния клас `^[0-9A-Z]` съвпада с всяка цифра и главна буква в началото на низ. Символът `$` се използва за съвпадение с низове, завършващи с дадена маска. Фигурните скоби (`{}`) се използват за откриване на многократни повторения на символните класове. Основните функции за регулярни изрази са `ereg()` и `eregi()`, които връщат положително число еквивалентно на `true`, ако маската бъде открита в изходния низ или празна стойност еквивалентна на `false` в противен случай. Ние сме използвали функцията `eregi()`, тъй като при нея не е от значение големината на буквите, когато се търси съвпадение.

Ако всички, въведени от потребителя, данни са валидни и не съществуват вече регистриран потребител със желаното от потребителя име и е – mail адресът му е различен от този на администратора, тогава се прави обръщение към базата данни и се прибавя ред в таблицата USERS. Обръщението към базата данни се прави с функцията `$row=DB_query(61,$userData)`; Тази функция е дефинирана в библиотеката “`setup_mysql.php`” и има два входни параметъра. Първият е `$n` – номер на заявката, а втория данните които са свързани с заявката. В този случай е масив с данните на потребителя. При успешно прибавяне на ред в таблицата, регистрацията се счита за успешна и на екрана се генерира поздравително съобщение. Също така се проверява дали е разрешено изпращането на поща (това става в файла за настройки “`setup_options.php`” при дефиниране на променливата `$genEmailDisable - 0` означава, че е разрешено, а `1` - че не е разрешено). Ако е разрешено изпращането на поща, то правим това чрез функцията `sendMail`, която е дефинирана в библиотеката “`functions.php`”:

```
$row=DB_query(61,$userData); // Добавя ред в таблицата USERS
if ($row) { //Ако наистина е добавен ред
if ($emailusers==1 and $genEmailDisable!=1){ // Ако е разрешено изпращането на поща
    $emailMsg=ParseTpl(makeUp('email_user_register')); //Генерира се текста на
    писмото,като се използват шаблони
    $sub=explode('SUBJECT>>', $emailMsg); $sub=explode('<<', $sub[1]);
    $emailMsg=trim($sub[1]); $sub=$sub[0];
    sendMail($userData[4], $sub, $emailMsg, $admin_email, $admin_email); // Изпраща се
    писмото
}
.....
//----->

function sendMail($email, $subject, $msg, $from_email, $errors_email) {
global $genEmailDisable;
if (!isset($genEmailDisable) or $genEmailDisable!=1){
    $msg=str_replace("\r\n", "\n", $msg);
    $php_version = phpversion();
    $from_email = "From: $from_email\r\nReply-To: $errors_email\r\nErrors-To:
    $errors_email\r\nX-Mailer: PHP ver. $php_version";
    mail($email, $subject, $msg, $from_email);
```

```
}  
}
```

За да изпратим поздравително текстово съобщение по електронна поща, използваме вградената команда на PHP `mail()`, където `$email` е адреса на получателя, `$subject` е тема на съобщението, `$msg` е тяло на съобщението и `$from_email` е низ, който се добавя в края на заглавието на съобщението и ако е на няколко реда, то те се разделят с “\r\n”. За доставка на съобщението тази команда разчита на локална система за електронна поща и за това е необходимо да укажем на PHP инсталацията с коя програма за изпращане на поща ще работим. Ако такава локална система (пощенски сървър) не е налична, то може да се забрани изпращането на поща от файла за настройки. Пощенските сървъри поддържат два основни протокола за четене на пощенски кутии на потребители: POP3 (Post Office Protocol версия 3) и IMAP (Internet Message Access Protocol). Нито един от тези протоколи не е предназначен за изпращане на поща – за тази цел използваме SMTP (Simple Mail Transfer Protocol). За използваме функцията `mail()` е необходимо да имаме инсталирана библиотеката IMAP. Инсталирането на SMTP сървър е обяснено в Приложение 1 от настоящата дипломна работа.

За да влезем с вече регистрирания потребител можем да изберем препратката начало от менюто на форума. При въвеждане на потребителско име и парола и натискане на бутона Вход правим отново обръщение към файла “`index.php`”, където променливата `$mode` има стойност `login`, а променливата `$action` не е указана. Прави се проверка дали дадения потребител съществува в базата данни. Ако съществува такъв потребител тогава се проверява неговата парола. Тъй като тя е съхранена в криптиран формат се използва вградената функция `md5()`, която за един и същ низ винаги връща една и съща поредица от символи. Ако подадената парола е същата като запазената в базата данни, тогава се създава сесия за този потребител.

```
if ($mode == 'login') {  
    $user_usr=trim($user_usr);  
    .....  
    else {  
        if ($row = DB_query(1,0))  
        {
```

```
// Потребителя съществува в базата данни, проверяваме неговата парола
$username = $row[0]; $userpassword = $row[1];
if ($username == $user_usr and $userpassword == md5($user_pwd))
{
    $logged_user = 1;
    $cook = $user_usr."|".md5($user_pwd)."|".$cookieexptime;
    %Изтриване предишната стойност на сесиината променлива $cookieName
    setcookie($cookieName,"",(time() - 2592000),$cookiepath,$cookiedomain,$cookiesecure);
    setcookie($cookieName, $cook, $cookieexptime, $cookiepath, $cookiedomain, $cookiesecure);
    Установяването на сесия (бисквитка) става с вградената функция:
    setcookie($cookieName, $cook, $cookieexptime, $cookiepath, $cookiedomain,
    $cookiesecure) ,където:
```

- \$cookieName е името на сесиината променлива (името се указва от файла за настройки);
- \$cook е стойността на сесиината променлива, която ние строим като комбинация от името на потребителя, неговата парола в криптиран формат и датата, след която бисквитката няма да е валидна.
- \$cookiepath е път, който указва къде ще валидна бисквитката
- \$cookiedomain е адрес, където е валидна бисквитката
- \$cookiesecure е указва дали ще се изпрати бисквитката, ако HTTP връзката не е сигурна

След като един потребител е влязъл в страниците на форума, то темите и форумите се показват сортирани във вида поискан от него при регистрацията. Също така при добавяне на тема или отговор, благодарение на стойността на сесиината променлива, можем да укажем на SQL заявката кой е автор на тема или отговора.

Тъй като административната зона има публичен адрес <http://traditions-bg.com/forum/admin.php> (въпреки, че до него няма директна препратка от никъде другаде във форума) не може да се гарантира, че някои злонамерени Интернет потребители не биха се опитали да го намерят или да имат достъп до него. Тъй като тези административни страници се използват за поддръжка и редактиране на

съдържанието на форума, те трябва да бъдат защитени от неоторизиран потребителски достъп. Затова достъп до тях е възможен след въвеждане на потребителското име и парола на администратора. По този начин се гарантира, че само администратора ще има достъп до административната зона. Това е реализирано със същата схема описана по – горе и използвана при влизането на обикновен потребител във страниците на форума.

Излизането на даден потребител от форума се осъществява отново от файла “index.php”, където променливата \$mode предавана от HTML формите има стойност logout. Изходът на потребителя се осъществява като изтриваме сесията променлива. Това става чрез повторно извикване на функцията setcookie() , като ѝ подаваме същото име и срок на валидност в миналото.

```
if($mode == 'logout') {  
    setcookie($cookie_name,"(time () - 2592000),$cookie_path,$cookie_domain,$cookie_secure);
```

Препратката, която дава възможност за управление на форума е видима и се намира в долната част на всяка страница, в зависимост от това дали администраторът е влязъл в страниците на форума със своето потребителско име и парола, които са зададени в файла за първоначални настройки “setup_options.php”.

- **Генериране на различните HTML страници в зависимост от действието указано от потребителя. Използване на шаблони.**

Web документите в нашия форум се генерират посредством HTML. За описание на външния му вид са използват каскадни набори от стилове (CSS), а за генериране съдържанието му се използва PHP. Съществува идея за разделяне на външния вид от съдържанието и тя може да бъде разширена и до скриптовете. Сайтовете са по – лесни за поддържане, ако се разделят тяхната логика, съдържание и външност. Това означава да разделим PHP и HTML код. Ако не се направи това с нарастването на даденото приложение ще е много трудно да се смени дизайна му. За това ние използвахме система за шаблони, която включва статични HTML шаблони в динамичните PHP файлове. Основното предимство е, че този който създава дизайна може въобще да не познава PHP кода. Създаването на тези шаблони подробно е описано в дипломната работа на Камелия Косева. Тук ще бъде описан начина, по който те се интегрират в динамичния код.

За целта използваме функцията `makeUp($name)`, която връща като резултат регулярен израз зареден от файл, който се намира в директорията `“/forum/templates”`.

```
function makeUp($name) {
    global $l_meta, $indexphp;
    if (substr($name, 0, 5)=='email') $ext = 'txt'; else $ext = 'html';
    if (file_exists ('./templates/'.$name.'.'.$ext)) { //ако файл с търсеното име съществува
        $fd = fopen ('./templates/'.$name.'.'.$ext, 'r'); //този файл се отваря за четене
        $tpl = fread ($fd, filesize ('./templates/'.$name.'.'.$ext)); //взима се съдържанието на файла
        fclose ($fd);
    }
    else die ("FATAL: NOT FOUND $name");
    return $tpl; // съдържанието на файла се връща като резултат
}
```

Параметърът `$name` е името на шаблона, който желаем да заредим. Ако в името му се съдържа низа `“email”`, тогава файла който търсим е с разширение `txt`, в противен случай е `html`. Когато търсим текстов файл функцията с извиква, за да се генерира съобщение за изпращане на писмо (примерно поздравително писмо за успешна регистрация на потребител). Върнатият резултат от функцията `makeUp()` се обработва допълнително от функцията `parseTpl ()`. Примерно извикване е `$loginLogout=ParseTpl(makeUp('user_login_form'))`, което ще върне генериран HTML код за формата за влизане в страниците на форума.

```
function parseTpl($tpl){
    $qs=array();
    $qv=array();
    $ex=explode ('{',$tpl);
    for ($i=0; $i<=sizeof($ex); $i++){
        if (!empty($ex[$i]) and substr_count($ex[$i],'}')>0) {
            $xx=explode('}',$ex[$i]);
            if (substr_count($xx[0],'{')>0) {
                $clr=explode ('{',$xx[0]); $sp=$clr[1]+0; $clr=$clr[0];
                if (!in_array($clr,$qs)) {$qs[]=$clr; global ${$clr};}
                $to=${$clr}[$sp];
            }
        }
    }
}
```

```
else { if(!in_array($xx[0], $qv)) {$qv[]=$xx[0]; global ${$xx[0]};}
$to=${$xx[0]};
}
$tpl=str_replace('${.$xx[0].}', $to, $tpl);
}
}
return $tpl;
}
```

HTML шаблона за влизане във форума изглежда по следния начин:

```
<span class=txtSm>
<b>&raquo; {$_sub_name}</b>
<input type=text size=10 maxlength=40 name="user_usr" value="{$_user_usr}" class=textForm
tabindex=3>
<b>&raquo; <a href="{$_indexphp}action=sendpass">{$_sub_pass}</a>
</b> <input type=password size=10 maxlength=32 name="user_pwd" class=textForm
tabindex=4></span>
<input type=hidden name=mode value=login>
```

Функцията ParseTpl заменя всяка една от променливите заградени във фигурни скоби (например {\$_sub_pass}) с реалната ѝ стойност, която е дефинирана в библиотеката “bul.php” или “eng.php” (\$_sub_pass='Парола'). Обработеният код от функцията ParseTpl изглежда по следния начин:

```
<span class=txtSm>
<b>&raquo; Вход</b>
<input type=text size=10 maxlength=40 name="user_usr" value="" class=textForm tabindex=3>
<b>&raquo; <a href="index.php?action=sendpass">Парола</a>
</b> <input type=password size=10 maxlength=32 name="user_pwd" class=textForm
tabindex=4></span>
<input type=hidden name=mode value=login>
```

В зависимост от това на кой език е зареден форума се включва съответната библиотека. По този начин е реализирана смяната на езика в страниците на форума. Смяната на езика на самите теми не е възможна, тъй като не разполагаме с програма, която да превежда свободен текст от английски на български език и обратното. За това по изискване на нашия дипломен ръководител ст. ас. Евгения

Ковачева базата от данни, която съхранява информацията на форума е различна на двата езика.

Тъй като заглавната и крайната част на HTML е еднаква за всички страници, то те са обособени като функции. Основната идея при изграждането на шаблоните е, че дадена страница може да се раздели на множество логически части. На всяка от тези части е дадено име като 'user_login_form' от по – горният пример. Всеки шаблон съдържа произволен брой променливи, които се заместват с обикновен текст или с изходни данни от други шаблони.

- **Добавяне на форум**

Даден форум може да бъде добавен само от потребител с администраторски права. PHP файлът, който обработва заявките на администратора е “admin.php”. Подобно на “index.php” двете променливи \$mode и \$action, предавани от като скрити полета от HTML формите, указват какви действия трябва да бъдат извършени. Единственото, което е задължително да се добави нов форум е неговото заглавие. Използваме функцията дефинирана в библиотеката “setup_mysql.php” - DB_query(30,\$data). Всички останали събития предизвикани на потребителя и свързани с базата данни, се обработват от тази функция. Тези събития са следните: добавяне на потребител, форум, тема, отговор и уведомяване по е – mail; забраняване и разрешаване достъпът на дадено IP до форума; изтриване на потребител, форум, тема, отговор и уведомяване по е – mail, затваряне на тема и сортиране на темите по актуалност.

Потенциален проблем би възникнал, ако потребителят въведе във HTML формите специални символи като кавички (“”) или ъглови скоби. Те могат да предизвикат проблеми при изпращането на заявки към базата данни. PHP предоставя няколко функции за промяна на интерпретирането или маскиране на такива символи, като addslashes(), htmlspecialchars(), stripslashes(). Потребителят би могъл да въведе HTML, Java Script или друг код . Приемрно функцията htmlspecialchars() замества символите <, >, & и “ със следните \$lt;, >, & и ". Съдържанието на всяко въведено поле в HTML формите на форума преминава през тази обработка, за да може тези символи при нужда да бъдат взети

от базата данни и визуализирани по правилен начин в браузъра, без да им бъде правен допълнителен синтактичен анализ.

- **Разрешаване и отказване на достъп до форума.**

Разрешаването и забраняването на достъп на даден потребител е реализирано посредством таблицата BANNED, която съхранява IP адресите на забранените потребители. При всяко стартиране на форума се взима IP адреса на посетителя и се прави проверка съществува ли в таблицата със забранените адреси.

```
$thisIp = getIP());  
if (DB_query(89,$user_id)) {  
    $title = $_site_name." :: ".$_accessDenied;  
    echo ParseTpl(makeUp('main_access_denied')); exit;  
}
```

Ако резултатът е положителен на екрана се извежда подходящо съобщение уведомяващо потребителя за съществуващата забрана. Един IP адрес може да се забрани само от администратора и заявката се обработва от файла “admin.php”. Стойността на променливата \$action, която указва желаното от потребителя действие, е 'banUsr2' и се обработва по следния начин:

```
if (preg_match("/^[0-9.+]$/", $banip) and trim($banip)!=0) { //Ако IP адреса е валиден  
    $thisIp=$banip; $thisIpMask=array($banip,$banip);  
    $row = DB_query(89,0); //Проверка дали съществува дадения IP адрес в таблицата  
    if ($row) $warning = $_IpExists; else { //Ако съществува се извежда грешката  
        $row = DB_query(86,0); //Добавяне на IP адрес в таблицата  
        if ($row>0) $warning = $_IpBanned; else $warning=$_mysql_error;  
    } //Ако заявката не успее се извежда грешка за проблем с базата данни
```

- **Реализация на търсенето на потребители в страниците на форума**

Търсенето на потребител е достъпно от администраторския панел и може да се извършва по следните критерии: потребителско име, е – email адрес, идентификационен номер, потребители без съобщения и потребители без съобщения от дадена дата. Всеки от критериите е достъпен чрез радио бутон от HTML форма и търсенето във даден момент може да се извършва само по един

критерии. При първоначално стартиране на страницата за търсене променливата \$action има стойност 'searchusers' и зарежда HTML формата използвайки шаблона "admin_searchusers.html" със селектиран критерии идентификационен номер на потребителя.

```
case 'searchusers': //действието за обработка е зареждане на страницата за търсене
  $ci='checked'; //Радио бутон Идентификационен номер (ID) е селектиран
  $warning = ""; //няма съобщения за грешки
  $text2=ParseTpl(makeUp('admin_searchusers')); //зареждане на HTML шаблона
  break;
```

При реално натискане на бутона "търсене" действието, което се обработва се дефинира от низа 'searchusers2'. Променливата \$whatus съдържа низа, по който се извършва търсенето. В зависимост от стойностите на радио бутона се стартират различни заявки към базата данни. Те са дефинирани посредством функцията DB_query (\$n, \$sus). За всеки намерен резултат се генерира автоматично HTML код с помощта на функцията ParseTpl(\$Results).

6. Реализация на Flash анимацията.

Съществуват няколко документа, които е възможно да бъдат създадени с помощта на Flash и които спомагат дадено съдържание да бъде пренасяно.

Първият документ е FLA файл, в който се създава съдържание посредством вмъкване на текст, графики, аудио и видео файлове. Това е работния файл на Flash, в който се правят промени. Цялото съдържание се организира на сцена върху времедиаграма, която е поредица от кадри, или ключови кадри, които съществуват във времето и могат да са празни или пълни. Ключовите кадри са местата, където се случват промени във времедиаграмата като добавяне на ново съдържание или промяна на анимацията. Обикновените кадри се използват между ключовите за запълване на части от анимацията или времедиаграмата. Кадрите се адресират с по номер или по име посредством етикети на кадри. Кадрите могат да бъдат натрупвани един върху друг посредством добавянето на слоеве, които спомагат за разделяне на съдържанието или създаване на ефект на наслояване в документа. Те се намират също във времедиаграмата. Елементите на слоевете (графики и текст)

изглеждат един върху друг. Предимството при използване на множество слоеве е, че може да се променят елементите на един слой без това да се отразява на съдържанието на останалите слоеве. Други елементи, които се използват във работните Flash файлове са символите. Символите спомагат за намаляване на файловия размер, защото дават възможност за многократно използване на символа без това да води до увеличаване на размера на файла. Трите основни типа символи са филмови клипове, бутони и рисунки. Филмовия клип прилича на мини приложение, което от своя страна има времедиаграма, която работи независимо от главната времедиаграма. Символите са намират в библиотеката на Flash файла. Когато е необходимо да се използва даден символ се прави негова инстанция. Инстанцията е копие на символа и може да има много такива инстанции на сцената, производни на един и същ символ. Когато дадена графика се конвертирана в символ се намалява размера на файла, защото не е необходимо всеки път когато тя се “чете” да се възпроизвежда. Това вече е направено веднъж в библиотеката. Разликата между графичния символ и филмовите клипове е, че няма възможност графичният символ да бъде управляван посредством ActionScript, защото инстанциите му нямат имена. Графичните символи са полезни, когато са необходими статични графики в библиотеката. В уводната анимация са използвани само поредица графични символи, създаващи впечатление за движение при разглеждане в бърза последователност.

Съществуват два начина за създаване на специални движения и анимации във Flash - движение по междинни фази на фигури и покадрова анимация. Междинните фази се постигат чрез изчисленията, извършвани от Flash за анимиране, преместване или промяна на свойствата или формата на даден обект от SWF файла. Началото и края на дадено преобразование се задават с ключови кадри, а Flash създава самостоятелно движение, което запълва промените случващи се между двата кадъра. Покадровата анимация е по-традиционен начин за създаване на анимация, като ръчно се рисува всяка фаза на движението. Това позволява създаването на по – сложни ефекти. Анимациите от тип кадър по кадър обикновено отнемат повече време за създаване и увеличават размера на файла. В уводната анимация на сайта е използвана междинната анимация.

Flash поддържа два типа формати на изображения-векторни и растерни. Векторните графики са съставени от прави и криви линии и описания на техните свойства. Командите в една векторна графика указват на компютъра как да извежда линии и фигури, какъв цвят да използва, колко широки да са линиите и т.н. Растерните (bitmap) изображения са изградени от пиксели. Когато създава едно растерно изображение, се създава карта на разположението и цвета на всеки пиксел. Присъщият тип изображения за Flash са векторни. Всичко, което се чертае във Flash с помощта на инструментите за чертане, е векторен чертеж. Векторните графики имат важни предимства те са с малък размер на файловете и се мащабират, без това да се отразява на качеството на изображението. Растерните изображения не се създават от Flash и трябва да се използва външно приложение за това и после да се вкарат (импортират) във Flash. За разлика от векторните графики, растерните изображения не се мащабират добре (при увеличение губят качеството си). Елементарните растерни изображения обикновено са с по-големи файлове от съответните им векторни графики, но много сложните растерни изображения, като например снимки, често са по-малки от съпоставими векторни графики. За създаването на уводната анимация са използвани растерни изображения.

При създаването на работния FLA файл създадохме следните слоеве:

- **background** – в този слой е дефиниран фонът, на който се изпълнява анимацията. Той е един и същ за всички кадри от времедиagramата.
- **skrollimages** – в този слой е реализирано постепенното появяване и отдръпване на графичните елементи от сцената. Графичните елементи са вмъкнати в Flash библиотеката като графични символи, които се използват за създаването на инстанциите им на сцената. Прозрачността е реализирана със свойството на цвета прозрачност (Alpha), което има стойности от 0 (графиката е невидима) до 100 процента (графиката е напълно видима). За появяването и отдръпването на всяка картинка са използвани четири ключови кадри, в които единствената разлика е стойността на процента на прозрачност. Всяка двойка ключови кадри (с прозрачност на първия 0 процента и на втория – 100 процента и обратното) служи за дефиниране на началото и края на междинна анимация (Motion tween). Между всяка двойка ключови кадри са

позиционирани обикновени кадри, чрез които картинката се задържа на сцената, за да бъде разгледана.

- **text_bg** – в този слой е реализирано появяването и отдръпването на текста на български език, който описва графичните елементи, и е синхронизирано с тяхното появяване. За целта отново е използвана междинната анимация както в предишния слой с разликата, че текста е напълно видим. Flash притежава вградени възможности за работа с текст. Когато се използва статичен текст, Flash създава контури за всеки знак и ги използва за изобразяване на текста. Това води до увеличаване на SWF файла поради допълнителната контурна информация, която трябва да бъде включена. Този проблем може да се заобиколи по два начина. Първият е чрез използването на шрифтовете на устройствата, които използват системни шрифтове за дадения текст вместо очертания. Но това води до известен риск как ще изглежда текста при отделните потребители, тъй като Flash използва първия шрифт намерен в системата на потребителя. Другият начин, който е използван при нас е вмъкването на картинки с текст, които са предварително редактирани чрез Adobe Photoshop.
- **text_en** – в този слой е реализирано появяването и отдръпването на текста на английски език, който описва графичните елементи, и е синхронизирано с тяхното появяване.
- **line** – в този слой е реализирано появяването на разделителната линия между текстовете “Българските традиции” и “English traditions” в края на анимацията, като отново е използвана междинната анимация.
- **labels** – този слой не съдържа никаква графична анимация. Той се използва за именуване на кадри. Един кадър може да бъде адресиран посредством ActionScript чрез неговия номер или етикет. В този слой е сложен етикет “start” на кадър с номер три, от който реално започва анимацията. Използването на етикети за кадри е особено полезно при създаването на по големи Flash приложения и спомага за по – добрата ориентация във работният файл.
- **actions** – този слой също не съдържа никаква графична анимация. В него е поставен ActionScript кода, който е използван. Flash е базиран на

модел, който включва класове (classes), обекти (objects) и инстанции (instances). ActionScript включва определен брой вградени класове, представляващи комплексни типове данни, с помощта на които можем да накараме Flash да извършва различни задачи (Дехаан, 2004, стр.367). Тъй като не желаем анимацията да е циклична, а да спре да се възпроизвежда при достигане на последния кадър сме добавили функцията stop(); в последния кадър на слоя text_en. ActionScript код има дефиниран в три кадъра от работния файл.

- **preloader** – в този слой се управляват действията на анимацията в първите два кадъра, които имат съдържание само тук. Файлът с уводната анимация не е обменен файл. Зареждането му би отнело малко време дори при internet връзка със влошено качество. Но обикновено се създават така наречените “preloader” – и, които не стартират анимацията преди целият SWF файл да бъде свален на потребителската машина. Използван е следният Action Script код, който стартира реалната анимация чак когато целият SWF файл е зареден:

```
if (_root._framesloaded>=_root._totalframes) {// Когато всички кадри за заредени  
    gotoAndPlay("start"); //Стартира се възпроизвеждането на анимацията  
}
```

След като е създадено съдържанието, изходният работен файл, трябва да се експортира като филм на Flash. След експортиране файла като филм (с файлово разширение .swf), Flash го компресира и оптимизира така, че файлът става значително по-малък от оригиналния работен файл. Това става с функцията Публикуване. Там се избира формата, в които да се да бъде изходният файл и се задават настройките за този формат

Създаването на SWF файл е най-използвания за експортиране на филм на Flash. Това е първата стъпка по извеждането на филм в Web. Когато е поставен филм на Flash на HTML страница в този формат, потребителят може да го види чрез Web браузъра стига той да има инсталиран Flash Player. Филмът на Flash ще бъде възпроизвеждан поточно (streamed) по Web, което означава, че зрителите могат да започнат да го възпроизвеждат почти веднага, докато останалата част от филма продължава да се зарежда (изтегля от Web) във фонов режим.

7. Тестване, оценка и усъвършенстване на сайта.

За разработване на сайта за българските традиции е необходимо изучаване на поведението и начина на работа на различните браузъри, както и познания в областта на различните технологии като Hypertext Markup Language (HTML), скриптове, Cascading Style Sheets (CSS), web-сървъри, Бази данни и PHP.

Тестването на сайта бе направено на следните браузъри – Internet Explorer 5.0, Internet Explorer 5.5, Internet Explorer 6.0, Netscape Navigator 4.7, Netscape Navigator 6.0, Opera 7.03 и Mozilla FireFox 1.0. Тъй като CSS не се поддържа напълно от всички браузъри някой от дефинираните стилове не се визуализират коректно. Примерен проблем е визуализирането на дефинициите за цвят на скрол бара. Те се показват коректно единствено в Internet Explorer с версии по – високи от 5.0. Тествахме сайта на различни резолюции и монитори. Тъй като съдържанието е заключено в HTML таблици с фиксирана ширина не открихме проблем. Евентуално некоректно визуализиране на страниците би се получило и при минимизиране и максимизиране на прозореца на дадения браузър, ако съдържанието не е поставено в таблици. За коректното появяване на уводната анимация съответния браузър трябва да има инсталиран Flash Player. При тестването на различни компютри се установи, че при различните настройки на браузърите се получаваше разместване на шрифтовете и полетата във формите спрямо изображенията. Този проблем се реши след като фиксирахме големината на шрифтовете в пиксели.

Регистрацията и влизането на даден потребител с неговото име и парола във страниците на форума не функционира, ако на потребителската машина е инсталирана защитна стена (fire wall), която забранява сесиите. Голяма част отделихме за тестване дали данните въведени от потребителите са коректни. Например дали въведен валиден е – mail адрес. Версията на PHP трябва да е 4.0 или по – късна и да поддържа сесии.

Тестовите на разработката са извършени на Windows 98, Windows 2000, Windows XP, Apache/1.3.31, PHP/4.3.9, MySQL 3.29.49.

Web базирано представяне на традициите и обичаите на България

Оценката на сайта според нас е по – скоро положителна. Едни от главните предимства на сайта са :

- неговия дизайн – умелото подбиране на цветовата гама и графичните ресурси.
- лесна и интуитивна навигация.
- пълнота на информацията.
- бърза скорост на зареждане, благодарение на оптимизирането на снимковия материал.

Както всяка една разработка сайтът има и недостатъци. След направеното тестване и оценка разграничихме следните насоки, в които може да се работи за усъвършенстване на сайта:

- превод на наличната информация на други от широко разпространените езици – немски, френски, испански.
- могат да се добавят препратки към други сайтове с подобно съдържание
- би могло да се добави търсене по име, което да намира деня (празника), в който хората с това име празнуват именния си ден.
- може да се добави още снимков материал.
- смяната на езика би могла да е достъпна от всяка страница на сайта, не само от началната.

Сайтът за българските традиции и обичаи е достъпен от интернет адресът <http://traditions-bg.com> .

8. Заключение.

Разработеният сайт за българските традиции и обичаи е една малка част от по-голям проект, обхващащ всички аспекти от наследството на България. Основната цел на тази дипломна работа е да запознае читателя с оцелялото народно богатство на българския народ. Описани са голям брой от обичаите и традициите на българите от началото на миналия век - църковните празници, както и езическите обреди, запазили се и до днешни дни. Сайтът съдържа подробна информация на български и английски език, която се базира на голям брой изследвана книжна литература, както и информация, предложена в електронен вид.

За реализирането на настоящият сайт са използвани следните технически средства - HTML, CSS, Java Script, Flash, PHP и MySQL.

Особено внимание е обърнато на дизайна на сайта - избора на цветове, представяне на графичната информация и т.н, с които сайтът превъзхожда намерените в интернет пространството сайтове на подобна тема.

В началото на дипломната работа бяха поставени няколко цели и изисквания, на които се стремим да отговорим сайта и които смятаме, че успяхме да изпълним с настоящата разработка. В сравнение с останалите сайтове по аналогична тематика в българското и чуждестранното Интернет пространство може да се каже, че изготвеният сайт се нарежда на едно от първите места по атрактивност, пълнота и лесна навигация. Естествено сайтът има и недостатъци, които както и предимствата са посочени в предишната точка.

Въз основа на направения анализ и представеният проект се надяваме уважаемата комисия и рецензент да оценят труда ни и този на научната ни ръководителка по достойнство.

Искаме още веднъж да благодарим най-сърдечно на научната ни ръководителка Евгения Ковачева, на която държим изключително много.

9.Използвана литература.

1. Книга: Дехаан Йен (2004), Macromedia Flash MX 2004 – Официален учебен курс, СофтПрес; София.
2. Книга: Хесус Кастането,Хариш Роат, Саша Шуман, Крис Сколо, Дийпак Вилайът (2001), Професионално програмиране с PHP, СофтПрес; София.
3. Книга: Люк Уелинг, Лаура Томсън (2004), Разработване на проекти за Web s PHP и MySQL, СофтПрес; София.

10. Приложения.

10.1. Приложение 1.

В това приложение са обяснени първоначалните настройки на форума. Те се задават от файла “setup_options.php” и могат да се променят в зависимост от нуждите на локалната инсталация на форума.

- \$DB - име на активната база данни
- \$DBhost – име на хоста, на който работи сървър за база данни. Стойността по подразбиране е “localhost”
- \$DBname – име на активната база данни, в която са създадени таблиците на форума.
- \$DBusr – име на потребителя, който има разрешение за връзка към сървър за база данни.
- \$DBpwd – парола на потребителя.
- \$admin_usr – потребителско име на администратора
- \$admin_pwd – парола на администратора
- \$admin_email – e – mail адрес на администратора
- \$lang - език на форума ('bul' и 'eng'). В момента са налични два езика и те са български и английски. Библиотеките за тях се намират в папката “lang” на форума. За да се зададе нов език, трябва да се създаде нов php файл в папката “lang”, съдържащ превод на необходимите съобщения.
- \$main_url – съдържа URL адреса на форума. Примерно 'http://traditions-bg.com'.
- Имената на таблиците на форума са:
\$Tf='forums';
\$Tp='posts';
\$Tt='topics';
\$Tu='users';
\$Ts='send_mails';
\$Tb='banned';
и те трябва да бъдат еднакви с тези дефинирани във файла “traditions.sql”.

Web базирано представяне на традициите и обичаите на България

- \$genEmailDisable – указва дали е позволено изпращането на поща от страна на администратора на форума(0 – разрешено, 1 – не е разрешено)
- \$emailadmin – указва дали е позволено уведомяването на администратора за регистрацията на нов потребител във форума (0 - не е разрешено, 1 - разрешено)
- \$emailusers - указва дали е разрешено уведомяването на даден потребител при нов отговор на тема, от която той е заинтересуван (1 – разрешено, 0 – не е разрешено)
- \$sitename - указва името на сайта, към който е прикачен форума.

Първата стъпка преди стартирането на форума е създаването на базата данни и таблиците в нея (те са достъпни от файла “traditions.sql”). При първото стартиране на форума е необходимо да се въведат данните за администратора в базата данни. За целта трябва да се стартира файлът “\$main_url/forum/admin.php” и потребителя да влезе в системата на форума с потребителското име и парола зададени от променливите \$admin_usr и \$admin_pwd. След влизането като администратор е достъпна препратката “Възстанови информацията за администратора в базата данни!”, посредством която се извършва заявката.

Тези действия трябва да се извършат два пъти за езиците, на които е наличен форума.

10.2. Приложение 2.

1. Начална страница на сайта



2. Вътрешна страница на сайта



3. Картинна галерия на сайта



4. Карта на сайта



4. Разгъване на менюто



5. Под линкове на раздел от менюто

