



CRYPTOGRAPHY WARGAME

MISTIC: JOINT UNIVERSITY MASTER IN SECURITY
OF INFORMATION AND COMMUNICATION TECHNOLOGIES

Master thesis report for the studies of Joint University Master in Security of Information and Communication Technologies presented by Ramon Fuguet Roma and directed by Cristina Pérez Solà.

Universitat Oberta de Catalunya, Barcelona, 2013

Abstract

This document corresponds to the Treball de Final de Master (TFM) of Universitat Oberta de Catalunya. The chosen project is the creation of a platform of Cryptography Wargames. The main goal of the project is to design and develop a platform to host cryptography wargames, that is, wargames focused on solving cryptographic codes. The project includes the creation of a front-end where the users can play cryptography quizzes, and the creation of a back-end where administrators can manage the platform. It also includes the tools to secure the platform against the most common web attacks, and also some example quizzes.

Contents

1	Introduction	1
1.1	Goals	1
1.2	Methodology	2
1.2.1	Requirement Analysis	2
1.2.2	Design	2
1.2.3	Construction	2
1.2.4	Verification	2
1.3	Planning	2
1.3.1	PAC1: Planning (27-Febr to 18-Mar)	3
1.3.2	PAC2: Design and Construction (18-Mar to 22-Apr)	3
1.3.3	PAC3: Construction (22-Apr to 31-May)	4
1.3.4	PAC4: Testing (31-May to 14-Jun)	4
1.3.5	Virtual presentation (14-Jun to 21-Jun)	4
1.3.6	Time Planning	4
1.4	Outline	5
2	State of Art	7
2.1	Project Euler	7
2.2	Hack this Site	8
2.3	Euskal Encounter HackIt	8
2.4	CodingBat	8
2.5	Over the Wire	9

3	Analysis	11
3.1	Identify Users	11
3.1.1	Guest	11
3.1.2	Player	11
3.1.3	Administrator	11
3.2	Functional Requirements	11
3.2.1	Guest	12
3.2.2	Player	12
3.2.3	Administrator	12
3.3	Security Requirements	13
3.3.1	Access control	13
3.3.2	Safe register	13
3.3.3	Web Attacks	14
3.3.4	Password protection	14
3.3.5	Prevent bruteforcing	14
3.4	Use cases	14
3.5	Other requeriments	14
4	Design	17
4.1	Database Model	17
4.1.1	Wargame	17
4.1.2	Quiz	18
4.1.3	Wargame Quiz	18
4.1.4	User	19
4.1.5	Quiz User	19
4.1.6	Level	20
4.2	Design Patterns	20
4.2.1	MVC	20
4.2.2	DAO	20
4.2.3	DTO	20
4.2.4	OOP	21
4.3	User Interface Design	21

5	Final Product	25
5.1	Front-end	25
5.1.1	Home	25
5.1.2	Register	26
5.1.3	Users	28
5.2	Back-end	29
5.2.1	Access	29
5.2.2	Wargames management	29
5.2.3	Quizzes management	30
5.2.4	Users management	31
5.2.5	Levels	32
5.2.6	Stats	33
5.3	Tools	33
5.3.1	Development	33
5.3.2	Other Tools	34
5.4	Example quizzes	35
5.4.1	Message in a column	35
5.4.2	The cursed book	35
6	Conclusion	37
6.1	Conclusion	37
6.2	Future work	38
6.2.1	More content	38
6.2.2	Dynamic quizzes	38
6.2.3	Users can send quizzes	38
6.2.4	Achievements	38
	Bibliography	39
	Appendices	39
A	Installation, Requirements and Deployment	41
A.1	Installation	41
A.1.1	Database	41
A.1.2	Code	41
A.2	Software requirements	42
A.3	Deployment	42

List of Figures

1.1	Time planning	4
2.1	Project Euler	7
2.2	Hack This Site	8
2.3	Euskal Encounter 2012	8
2.4	CodingBat	9
2.5	Over the Wire	9
3.1	Use cases	15
4.1	Database Model	17
4.2	Quizzes List	21
4.3	Solving a Quiz	22
4.4	User profile	22
4.5	Backend: Quiz editor	23
4.6	Backend: User list	23
5.1	Home page	25
5.2	Register to the platform	26
5.3	Login	26
5.4	Wargame list	27

5.5	Quiz list	27
5.6	Solve quiz	28
5.7	Users list	28
5.8	User profile	29
5.9	Access to the back-end	29
5.10	List of wargames	30
5.11	Editing a wargame	30
5.12	List of quizzes	31
5.13	Editing a quiz	31
5.14	Users list	32
5.15	Edit user	32
5.16	Levels list	32
5.17	Edit level	33
5.18	Statistics	33
5.19	Starting level: Message in a column	35
5.20	Second level: The cursed book	36

CHAPTER 1

Introduction

This document corresponds to the *Treball de Final de Màster (TFM)* of *Màster Interuniversitari en Seguretat de les TIC (MISTIC)* of *Universitat Oberta de Catalunya*. The chosen project has been the creation of a platform of Cryptography Wargames.

A wargame is a security challenge in which one must exploit a vulnerability in a system or application or gain access to a computer system. These challenges can also involve programming or mathematic quizzes.

In this project, we design and develop a wargame platform whose topic will be cryptography. In these wargames, the challenges won't contain hacking problems, and instead will contain quizzes and cryptographic codes.

The platform is composed of two parts:

- A front-end where the users can register and solve quizzes
- A back-end where the administrators can manage the quizzes and users.

1.1 Goals

The main goal of this TFM is to develop a platform to manage cryptography wargames. The minimum requirements are:

- Create a front-end that allows the users to play cryptography quizzes in different wargames.
- Create a back-end where the administrators can manage wargames, quizzes and users.

- Analyze and develop a strong defense system against the most popular web attacks.
- Develop a test wargame that contains some quizzes based on some easy cryptographic systems.

1.2 Methodology

In order to develop this project, I have used the Waterfall methodology, a sequential design process, in which progress is seen as flowing steadily downwards through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production and Maintenance. This model organizes each of the phases in such a way that the start of a phase has to wait for the ending of the previous one. In this case the phases are:

1.2.1 Requirement Analysis

The needs of the end users are analyzed, in order to decide what objectives have to be covered. In this phase, requirement specification document is created, that contains the complete specification without providing internal details. In this case, the needs of the project are provided by the consultant teacher.

1.2.2 Design

The system is divided into different elements to develop, and these are structured in tasks. The modules and their relations are identified. Finally, the code organization and the main design patterns are decided.

1.2.3 Construction

The solutions proposed in the design phase are coded, and the product is built. The construction also includes a test phase.

1.2.4 Verification

The end user uses the software, and executes an acceptance plan, in order to check if the product satisfies all the proposed objectives.

1.3 Planning

The project is divided in 4 PACs (Ongoing Evaluation).

1.3.1 PAC1: Planning (27-Febr to 18-Mar)

In this first phase, the project is planned according to the needs of the product and the time available for every phase. The main tasks are:

- Study the problem to solve
- Identify the main objectives
- Choose a work methodology
- Define the project scope
- Break down the project scope into tasks
- Time planning

Finally, with this information a SOW (Statement of Work) document is created.

1.3.2 PAC2: Design and Construction (18-Mar to 22-Apr)

The tasks of the second phase are:

- Analyze and create the database model
- Front-end: Wargames list
- Front-end: Wargame visualization (List Quizzes)
- Front-end: View and solve a quiz.
- Back-end: Wargame management (Create, Read, Update, Delete)
- Back-end: Quiz management (Create, Read, Update, Delete, Organization)
- User Management: Register new user
- User Management: Login, Logout
- User Management: Recover password
- User Management: View and edit a user profile
- Start the report of the project

1.3.3 PAC3: Construction (22-Apr to 31-May)

The tasks of the third phase are centered in closing the construction phase:

- Front-end: Welcome page
- Front-end: FAQ
- Front-end: Bruteforcing control when solving quizzes
- Front-end: Users list organized by score
- Front-end: User statistics
- Back-end: Login to back-end with admin permissions
- Back-end: User management (Create, Read, Update, Delete, Search, Blocking)
- Back-end: Platform statistics
- Content of the quizzes
- Developing the report of the project

1.3.4 PAC4: Testing (31-May to 14-Jun)

In the fourth phase the final product and the report are delivered.

- Integration testing of the product
- Finish the report of the project

1.3.5 Virtual presentation (14-Jun to 21-Jun)

In this final phase a virtual presentation is created. The presentation contains a demonstration of the functionality of the product.

1.3.6 Time Planning

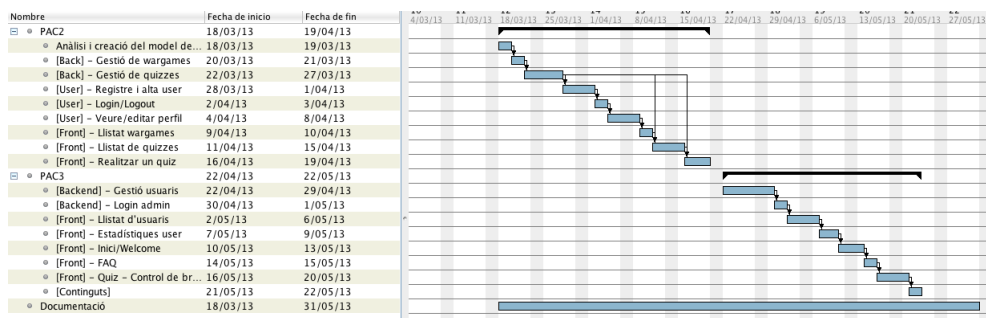


Figure 1.1: Time planning

Overall, the time planning has been followed and all tasks have been completed accordingly.

Nevertheless, there are a few tasks that needed more time than planned. In particular, I had to spend more time on user management tasks than I expected, and due to this, I have moved the documentation tasks to the second PAC. This has not been a problem because I always create a buffer task to prevent this kind of problem.

1.4 Outline

Chapter 1 introduces the objective of the project, and presents the main goals. It also contains the planning and the methodology used in this project.

Chapter 2 reviews the State of Art, analyzing different wargame platforms.

Chapter 3 shows the system analysis and the scope of the project.

Chapter 4 contains the information related to the technology used to develop the project, the data model and code structure.

Chapter 5 shows the final product, and describe the tools used to develop the project.

Chapter 6 exposes the conclusions.

CHAPTER 2

State of Art

Currently, there are several existing wargame platforms. Some are thematic, oriented to solve programming quizzes, mathematic exercises or hacking websites, and others are multi-purpose accepting different categories.

2.1 Project Euler

This site is specialised in mathematic quizzes using programming skills.

To solve the quizzes some mathematic knowledge is required, knowing at least one programming language and having logic skills.

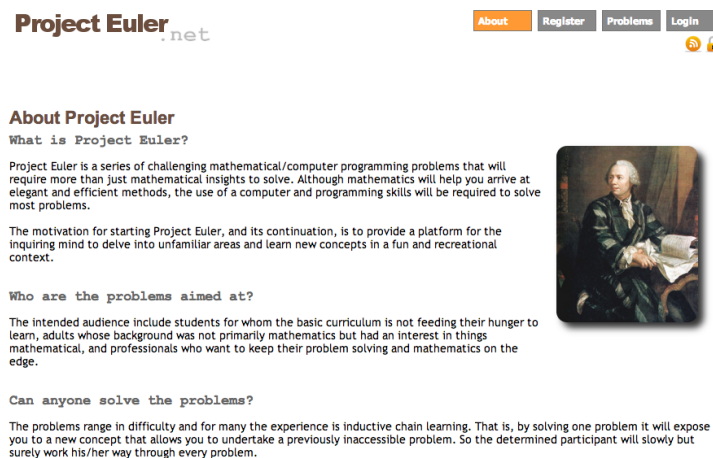


Figure 2.1: Project Euler

2.2 Hack this Site

Wargame platform that contains multiple challenges about intrusion, programming, cracking,...



Figure 2.2: Hack This Site

2.3 Euskal Encounter HackIt

Hacking quizzes and logic challenges that are launched every year during the event of Euskal Encounter.



Figure 2.3: Euskal Encounter 2012

2.4 CodingBat

It's a website more oriented to practice Java and Python than a real wargame platform, but the final levels of this site are challenging enough to consider it a wargame platform.

The screenshot shows the CodingBat website interface. At the top, there is a navigation bar with links for 'about', 'help', 'code help+videos', 'done', and 'prefs'. A login form is visible on the right with fields for 'id/email' and 'password', and buttons for 'log in', 'forgot password', and 'create account'. The main content is divided into two columns: 'Java' and 'Python'. Each column lists several 'Warmup' problems with star ratings and links to solutions. For example, under Java, there are 'Warmup-1' (5 stars) and 'Warmup-2' (4 stars). Under Python, there are 'Warmup-1' (4 stars), 'Warmup-2' (3 stars), and 'String-1' (3 stars). Each problem description includes a brief overview and a list of related problems with 'H' (hints) and 'more' links.

Figure 2.4: CodingBat

2.5 Over the Wire

It's a relatively new platform, but it reminds of old school wargames, where quizzes are about performing an intrusion to a server, get credentials, and hacking related problems.

The screenshot shows the OverTheWire website. The header features a navigation menu with 'News', 'Wargames', and 'About'. The 'Wargames' section is highlighted. Below the header, there is a list of wargames: Vortex, Semtex, Krypton, Bandit, Leviathan, Narnia, Behemoth, Utumno, Maze, and Marpage. The 'Wargames' section text states: 'The wargames offered by the OverTheWire community can help you to learn and practice security concepts in the form of fun-filled games. To find out more about a certain wargame, just visit its page linked from the menu on the left. If you have a problem, a question or a suggestion, you can join us on IRC.'

Figure 2.5: Over the Wire

3.1 Identify Users

3.1.1 Guest

Guest is a visitor user that is not logged in to the platform. The actions that a guest user can perform are limited to visiting some sections of the front-end and viewing the available wargames. A guest user cannot see the content of the wargames or play any quiz.

3.1.2 Player

When a visitor user registers and logs in, they become a player user. A player user can visit all the sections of the front-end and solve quizzes. They cannot access the back-end.

3.1.3 Administrator

An administrator is a user with complete access to front-end and back-end. As administrator, they can manage wargames and quizzes, adding news, editing or deleting them if necessary. They can also manage users, editing or blocking them.

3.2 Functional Requirements

The functional requirements for every user are detailed next.

3.2.1 Guest

A guest user can only access the front-end and perform the following actions:

- View homepage
- View wargames list
- View users list
- View details of user
- View FAQ page
- Register to the platform
- Login to the platform
- Recover password

3.2.2 Player

A player user can access the front-end and perform the same actions as a guest user and additional actions:

- View quizzes lists
- View the content of the quizzes
- Play quizzes
- Edit his profile
- Upload an avatar
- Logout from the platform

3.2.3 Administrator

An administrator user can access the front-end and the back-end and perform the same actions as a player user and additional actions:

- View platform stats
- Create wargame
- Edit wargame
- Delete wargame

- Create quiz
- Edit quiz
- Delete quiz
- Add quiz to a wargame
- Remove quiz from a wargame
- Organize quizzes in a wargame
- Enable / Disable a quiz from front-end
- Search users by name and email
- Create user
- Edit user
- Delete user
- Activate user
- Block user
- Add admin privileges to a user
- Create level
- Edit level
- Delete level

3.3 Security Requirements

3.3.1 Access control

The platform has to be programmed to be able to verify that a user cannot access the forbidden sections, and to control access.

3.3.2 Safe register

Register process, login action and recovery password process are sections that are especially sensitive in case of suffering an attack. These actions must have verifications in order to avoid identity theft.

3.3.3 Web Attacks

The platform has to be resistant against the most common web attacks including:

- Sql Injection
- XSS (Cross-site scripting)
- Username enumeration
- Cross Site Request Forgery

3.3.4 Password protection

The access to the platform is performed via password, and this must be protected. When a new user is registered, he must be able to choose a password and the system has to verify its complexity, and only accept it if it meets the requirements.

The password must be coded in order to avoid password leaking and then be stored in the database.

3.3.5 Prevent bruteforcing

Some of the users can feel tempted to use a kind of bruteforcing mechanism to solve the quiz. There has to be a control in order to avoid this technique.

3.4 Use cases

3.5 Other requirements

- One wargame can contain multiple quizzes and one quiz can belong to more than one wargame
- Quizzes can be organized by the administrator
- There are statistics about the users and quizzes
- The quizzes have to allow multiple style options (bold, italic, font-size, etc)

Uses Cases



Figure 3.1: Use cases

4.1 Database Model

This diagram shows all the tables of the platform and their relations. The functionality of these tables and their fields are detailed next.

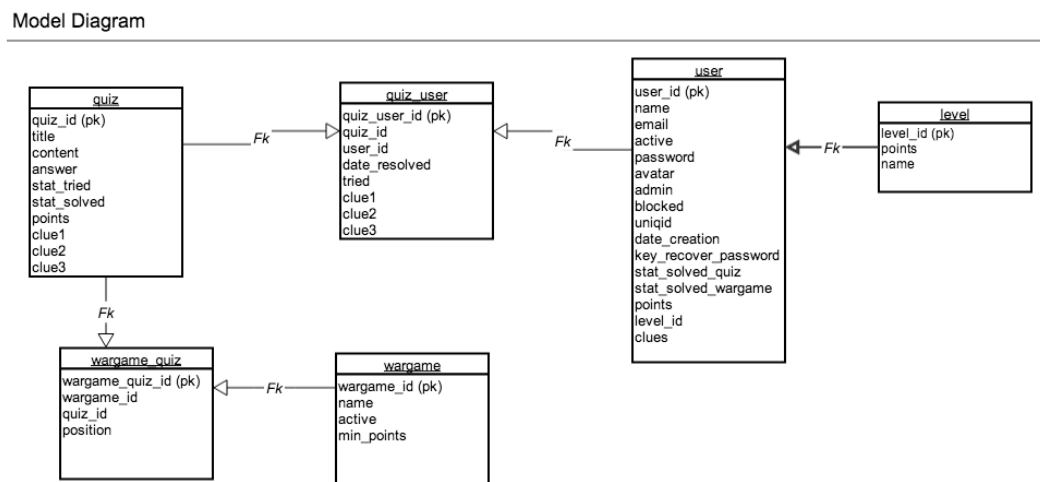


Figure 4.1: Database Model

4.1.1 Wargame

This table contains the information of the wargames.

- `wargame_id`: Primary Key
- `name`: Title of the wargame that will be shown in the list
- `active`: Controls if the wargame will visible in the frontend.
- `min_points`: Number of points that have to have a user in order to access this wargame

4.1.2 Quiz

This table contains the information of the quizzes, and some stats related.

- `quiz_id`: Primary Key
- `title`: Title of the quiz that will be shown
- `content`: Content generated by the Rich Text Editor
- `answer`: Answer of the quiz
- `stat_tried`: Number of attempts that all users have tried to solve it
- `stat_solved`: Number of users that have solve the quiz
- `points`: Points that a user will win if can solve the quiz
- `clue1`: Clue to help users to solve this quiz
- `clue2`: Clue to help users to solve this quiz
- `clue3`: Clue to help users to solve this quiz

4.1.3 Wargame Quiz

This table relates the quizzes with the wargames. The relation is N-M: One wargame can contain multiple quizzes and one quiz can be shown in multiple wargames.

- `wargame_quiz_id`: Primary Key
- `wargame_id`: Foreign Key to Wargame Table
- `quiz_id`: Foreign Key to Quiz Table
- `position`: Position inside the wargame where the quiz will be ordered.

4.1.4 User

This table contains the information of the users (players and administrators) registered in the platform. This table also contains stats user related, and fields used by user management processes.

- `user_id`: Primary Key
- `name`: Name of the user
- `email`: Email registered
- `active`: If the user has been activated via email confirmation
- `password`: Password of the user (coded)
- `avatar`: Filename of the avatar
- `admin`: If the user is admin
- `blocked`: If the user have been blocked by an administrator (for abuse or other reasons)
- `uniqid`: Id used in email confirmations in order to avoid showing the real `user_id`
- `date_creation`: Date when user was registered
- `key_recover_password`: This field is used to store a key when user lost his password. The link to recover the password is formed with this field and `uniqid`.
- `stat_solved_quiz`: Number of quizzes solved
- `stat_solved_wargame`: Number of wargames solved
- `points`: Total points achieved
- `level_id`: Level achieved
- `clues`: Number of clues the user can spend

4.1.5 Quiz User

This table relates the quizzes with the users. The relation is N-M: One user can solve multiple quizzes and one quiz can be solved by multiple users.

- `quiz_user_id`: Primary Key
- `quiz_id`: Foreign Key to Quiz table
- `user_id`: Foreign Key to User table
- `date_resolved`: Date when user solved the quiz. It is used to know if user have been solved the quiz or not.

- tried: How many times the user has tried this quiz
- clue1: For this quiz, if the user can view the clue 1
- clue2: For this quiz, if the user can view the clue 2
- clue3: For this quiz, if the user can view the clue 3

4.1.6 Level

This table contains the information of the levels that a user can achieve when he gets points.

- level_id: Primary Key
- points: Number of points that are required to get this level
- name: Name of the level

4.2 Design Patterns

4.2.1 MVC

Model–view–controller (MVC) is a software pattern which separates the representation of information from the user’s interaction with it. The application is divided into three kinds of components:

- Model: Is the representation of the information used by the system, and manages all the access (updates or queries) to this information.
- View: Is the component that shows the information in a suitable format, usually the User Interface.
- Controller: It responds to events, usually actions performed by users. This component acts between model and view sending commands.

4.2.2 DAO

Data access object (DAO) is a design pattern that provides an abstract interface to database or other persistence mechanism. By mapping application calls to the persistence layer, DAOs provide some specific data operations without exposing details of the database.

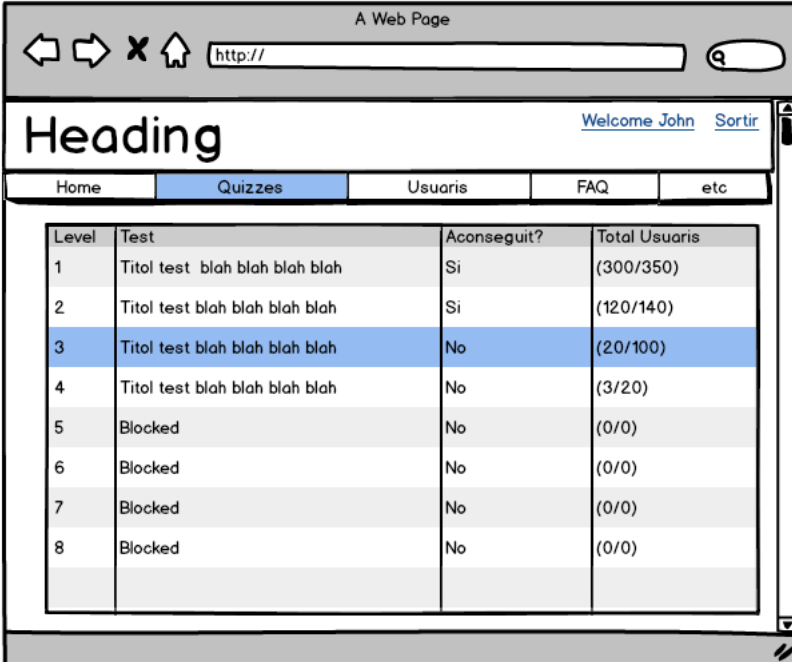
4.2.3 DTO

Data transfer object (DTO) is a design pattern used to transfer data between software application subsystems. DTOs are used in conjunction with data access objects to retrieve data from a database. DTOs are simple objects that should not contain any business logic that would require testing.

4.2.4 OOP

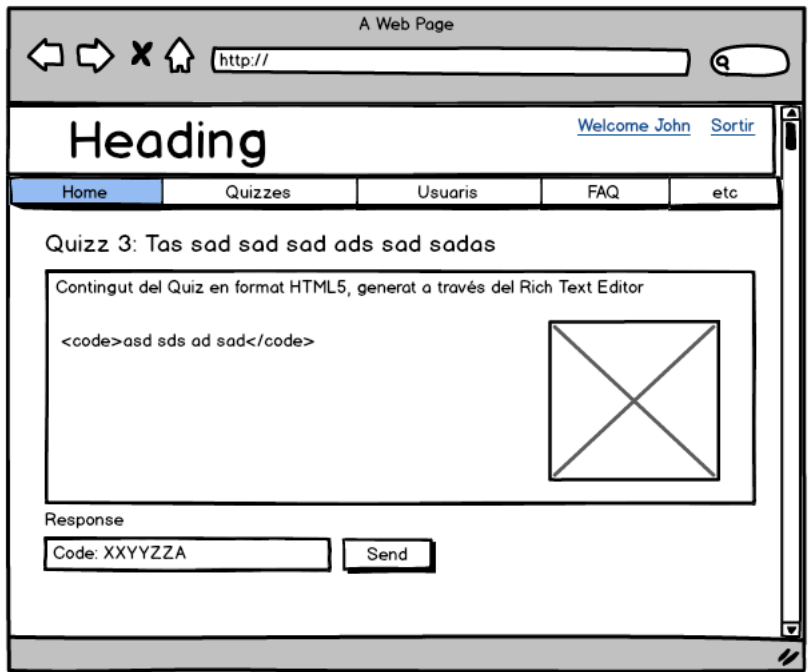
Object-oriented programming (OOP) is a programming paradigm that represents concepts as objects that have data fields (attributes that describe the object) and associated procedures known as methods. Objects are instances of classes and are used to interact with one another to design applications and computer programs.

4.3 User Interface Design



Level	Test	Aconsequit?	Total Usuaris
1	Titot test blah blah blah blah	Si	(300/350)
2	Titot test blah blah blah blah	Si	(120/140)
3	Titot test blah blah blah blah	No	(20/100)
4	Titot test blah blah blah blah	No	(3/20)
5	Blocked	No	(0/0)
6	Blocked	No	(0/0)
7	Blocked	No	(0/0)
8	Blocked	No	(0/0)

Figure 4.2: Quizzes List



created with Balsamiq Mockups -

Figure 4.3: Solving a Quiz



created with Balsamiq Mockup

Figure 4.4: User profile

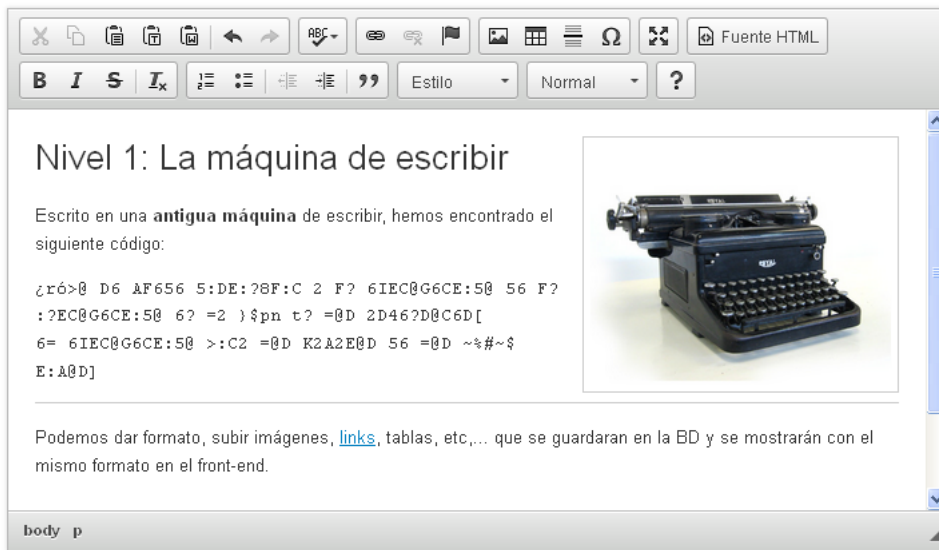


Figure 4.5: Backend: Quiz editor

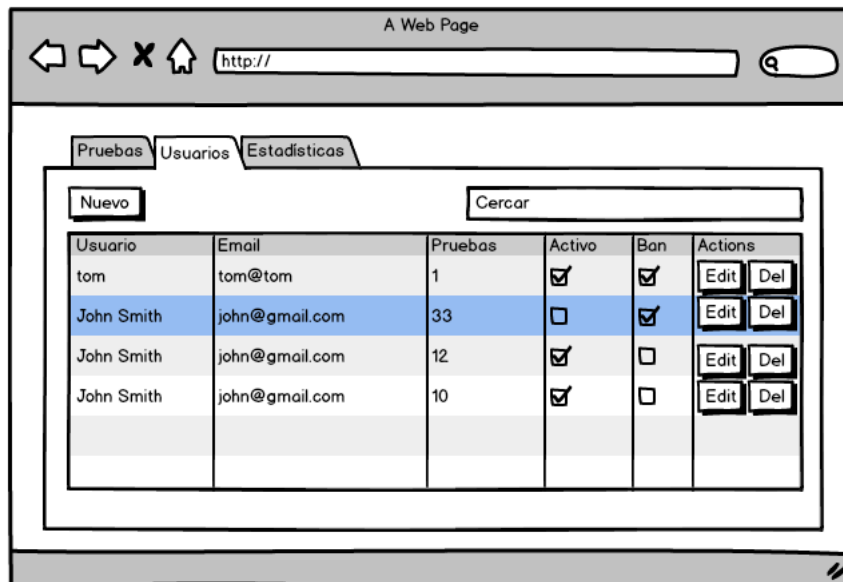


Figure 4.6: Backend: User list

Final Product

5.1 Front-end

5.1.1 Home

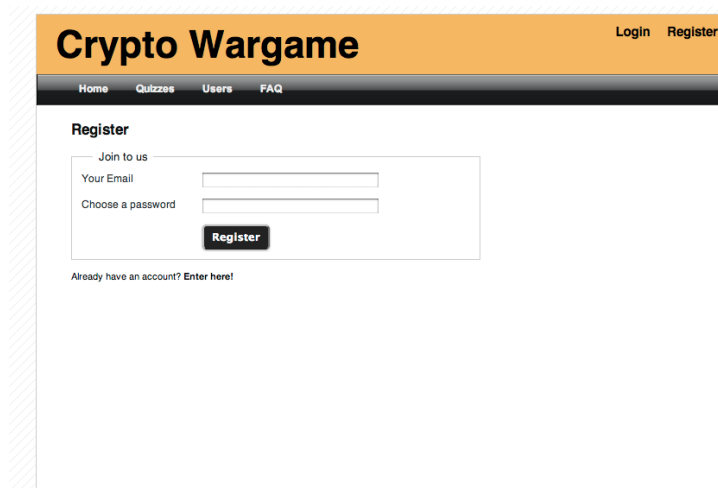
Home page offers a little explanation to the visitors about the site. More information is provided at the FAQ (Frequently Asked Questions) page.



Figure 5.1: Home page

5.1.2 Register

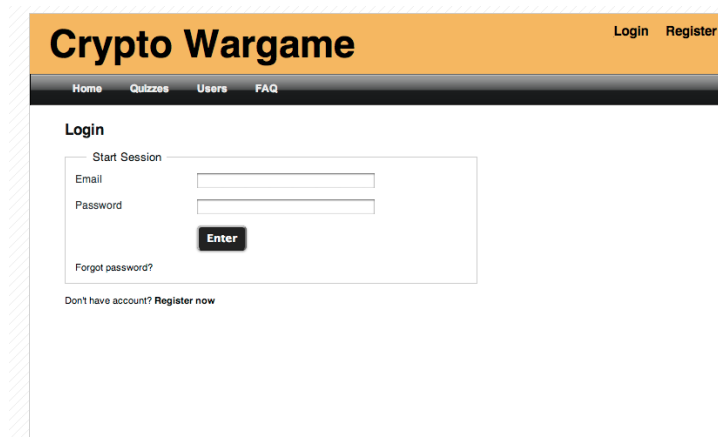
In order to play the quizzes, the users must register to the platform. The users have to provide a valid email address that is checked in the process of registering.



The screenshot shows the 'Register' page of the 'Crypto Wargame' platform. The page has an orange header with the site name 'Crypto Wargame' and links for 'Login' and 'Register'. Below the header is a navigation bar with links for 'Home', 'Quizzes', 'Users', and 'FAQ'. The main content area is titled 'Register' and contains a form with the following elements: a sub-header 'Join to us', a label 'Your Email' above an input field, a label 'Choose a password' above another input field, and a 'Register' button. Below the form, there is a link: 'Already have an account? Enter here!'.

Figure 5.2: Register to the platform

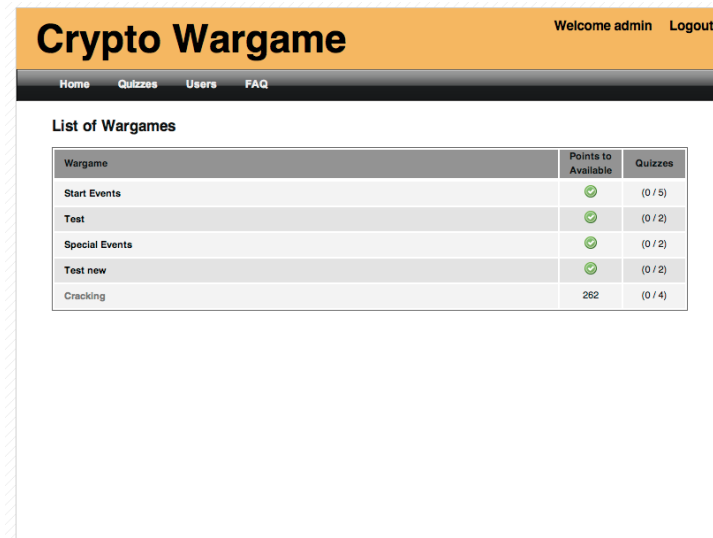
After registering, the users can log in, and access to the wargames and quizzes.



The screenshot shows the 'Login' page of the 'Crypto Wargame' platform. The page has an orange header with the site name 'Crypto Wargame' and links for 'Login' and 'Register'. Below the header is a navigation bar with links for 'Home', 'Quizzes', 'Users', and 'FAQ'. The main content area is titled 'Login' and contains a form with the following elements: a sub-header 'Start Session', a label 'Email' above an input field, a label 'Password' above another input field, and an 'Enter' button. Below the form, there is a link: 'Forgot password?' and another link: 'Don't have account? Register now'.

Figure 5.3: Login

In the front-end, all the active wargames are shown. Every wargame appears with the name, the total of quizzes it contains, and the number of quizzes the user has solved.



The screenshot shows the 'Crypto Wargame' admin interface. At the top, there is a navigation bar with 'Home', 'Quizzes', 'Users', and 'FAQ'. The main content area is titled 'List of Wargames' and contains a table with the following data:

Wargame	Points to Available	Quizzes
Start Events	✓	(0 / 5)
Test	✓	(0 / 2)
Special Events	✓	(0 / 2)
Test new	✓	(0 / 2)
Cracking	262	(0 / 4)

Figure 5.4: Wargame list

When accessing to a wargame, the list of quizzes is shown. The quizzes are organized in the same order that is provided in the backoffice. For every quiz the following information is shown:

- Title
- If the quiz has been solved by the user
- How many users have solved this quiz
- How many users have tried to solve this quiz



The screenshot shows the 'Crypto Wargame' admin interface for the 'Start Events' wargame. It displays a table with the following data:

Level	Test	Achieved	Stats
1	What is this code?	✓	(3 / 7)
2	Other theorem	✗	(2 / 20)
3	Caesar code	✓	(5 / 5)
4	Supercode	✗	(1 / 19)

Figure 5.5: Quiz list

By clicking on the quiz name, the user accepts to play this quiz. The content of the quiz is created at the back-end using a rich text editor, therefore it can contain a lot of different styles.

To solve the quiz, the player must enter the answer and click on solve quiz button. In order to avoid bruteforcing attacks, the form has a mechanism that controls the time from the last submit and if it is too short, shows an error message.

Figure 5.6: Solve quiz

5.1.3 Users

Finally, there is a ranking of users ordered by points scored.

Name	Points
admin	38 points
Lannister	5 points
Johnny	0 points
Number 1243	33 points
Joe	0 points
Joe 2	0 points
Stark	10 points
Adrian	0 points
Joe	5 points
Player1	0 points

Figure 5.7: Users list

And the user has the possibility to edit his profile and to upload an avatar.



Figure 5.8: User profile

5.2 Back-end

5.2.1 Access

In order to access the back-end it is necessary to enter a login and password of a user that has administrator privileges.

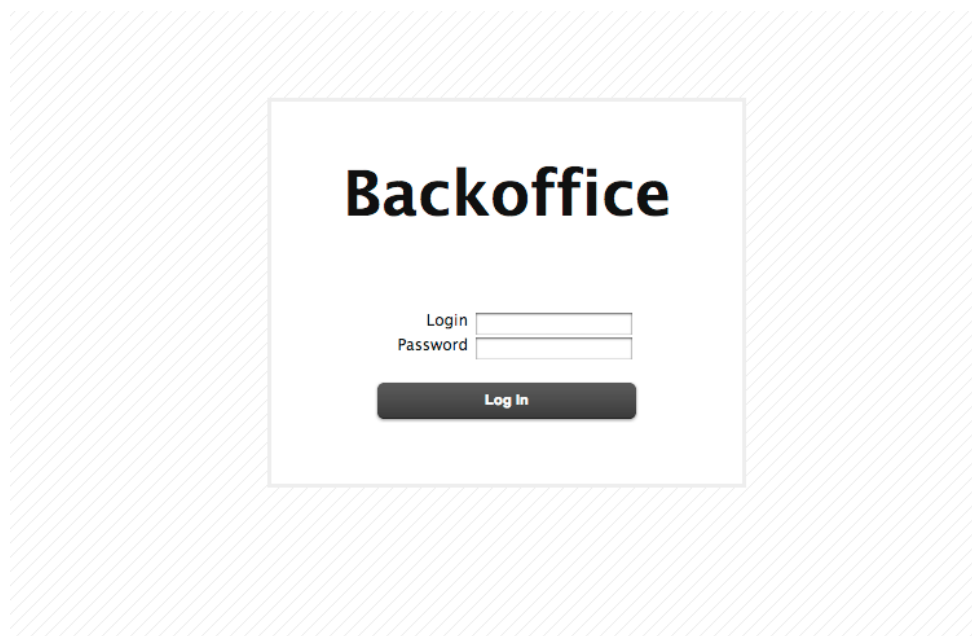


Figure 5.9: Access to the back-end

5.2.2 Wargames management

An administrator user can manage wargames, performing the following actions:

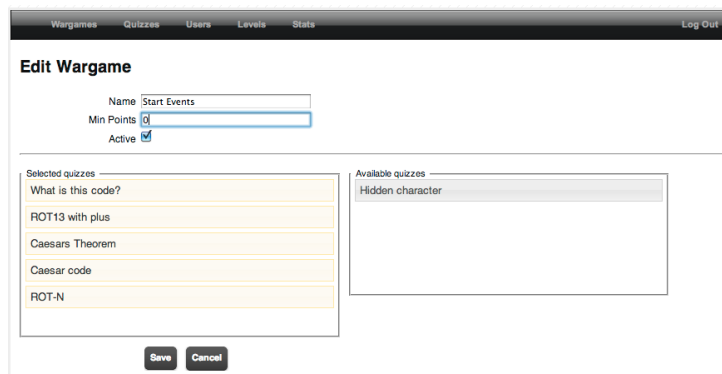
- Create
- Read
- Update
- Delete
- Add and remove quizzes to the wargame
- Organize quizzes inside the wargame



The screenshot shows a web application interface with a navigation bar containing 'Wargames', 'Quizzes', 'Users', 'Levels', 'Stats', and 'Log Out'. Below the navigation bar is a section titled 'Wargames' with a link to 'Add New Wargame'. A table lists several wargames with their respective details and actions.

Wargame	Min Points	Active	Actions
Start Events	0	✓	✎ ✖
Special Events	10	✓	✎ ✖
Encription	15	✓	✎ ✖
Cracking	300	✓	✎ ✖
Test	0	✓	✎ ✖
Coming soon	0	✗	✎ ✖
Pending to define	0	✗	✎ ✖
Test new	35	✓	✎ ✖

Figure 5.10: List of wargames



The screenshot shows the 'Edit Wargame' form. It includes fields for 'Name' (containing 'Start Events'), 'Min Points' (containing '0'), and 'Active' (checked). Below these fields are two lists: 'Selected quizzes' and 'Available quizzes'. The 'Selected quizzes' list contains 'What is this code?', 'ROT13 with plus', 'Caesars Theorem', 'Caesar code', and 'ROT-N'. The 'Available quizzes' list contains 'Hidden character'. At the bottom of the form are 'Save' and 'Cancel' buttons.

Figure 5.11: Editing a wargame

5.2.3 Quizzes management

An administrator user can manage wargames, performing the following actions:

- Create
- Read
- Update
- Delete

When creating or editing a quiz, the form shows a rich text editor that allows to use multiple styles. With that the quizzes can be created with more styles.

User	E-mail	Date Creation	Points	Active	Blocked	Admin	Actions
Blocked user	blocked@hotmail.com	2013-05-13 21:31:06	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
Not activated user	na@hotmail.com	2013-05-13 21:23:45	0	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Joe	joe@70bits.com	2013-04-25 21:35:58	6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 5.14: Users list

Figure 5.15: Edit user

5.2.5 Levels

An administrator can manage the levels that a user can obtain, performing the following actions:

- Create
- Read
- Update
- Delete

The level points are the points that a user needs in order to get achieved this level. When a user earns points, the system automatically checks if he has achieved a new level.

Level	Points	Actions
Hacker	500	
Guru	250	
Lord	200	
Expert	100	
Senior	75	
Smart Senior	75	
Senior	50	
Medior	50	
Starter	40	
Junior	30	

Figure 5.16: Levels list

Figure 5.17: Edit level

5.2.6 Stats

In the back-end there are several statistics that administrator can consult, like users registered, total quizzes solved, percentage quizzes solved,...

	Total
Total Wargames	7
Total Quizzes	5
Total Users	13
Quizzes solved	10
Quizzes tried	12
% solved	83.33 %

Figure 5.18: Statistics

5.3 Tools

5.3.1 Development

PHP

PHP is a server side scripting language oriented for web development. PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform. I have choosed PHP as a main language because it is a very popular language and this allows to deploy the application in the most common shared web hostings.

TWIG

Twig is a template engine for the PHP. It is known for be the default template engine of Symfony PHP framework. I have used this engine to separate de View from Controller and Model part in the MVC pattern.

Client language: Javascript + jQuery framework

JavaScript is an interpreted computer programming language. It was implemented as part of web browsers so client-side scripts can interact with the user. jQuery is the most popular javascript

framework, designed to simplify the client side scripting of HTML. The use of this library is make easier working with DOM (Document Object Model) elements, create animations, handle events and develop AJAX (Asynchronous JavaScript and XML) events.

In that case I have included this library as a requirement for the rich text editor.

Mysql

MySQL is one of the most used RDBMS (Relational Database Management System). It is a very popular choice of database for use in web applications, and it is installed in almost all web hostings.

Rich Text Editor: CKEditor

CKEditor is a WYSIWYG (What You See Is What You Get) text editor designed to be used in the web pages. I have used this editor for allow the creation of the quiz content with multiple styles.

5.3.2 Other Tools

IDE: Aptana Studio

Aptana Studio is an open source integrated development environment (IDE) for building web applications. It has support for PHP, Javascript, HTML, DOM, CSS, etc providing syntax coloring, code assist, syntax error annotations, code formatting, hyperlinking classes, and more features.

Development platform: MAMP

The MAMP is an acronym that refers to a set of free software programs commonly used together to run dynamic web sites on Mac OS X.

- M: Mac OS X, the operating system
- A: Apache, the web server
- M: MySQL, the database server
- P: PHP, the programming language

There are equivalents packages for the other operating system: WAMP for the Windows installation, LAMP for the Linux installation,...

Report: MacTeX

MacTeX is a distribution of TeX Live for the Mac OS X. MacTeX is pre-configured to work out-of-the-box with Mac OS X as it provides sensible defaults for configuration options that, in TeX Live, are left up to the user to allow for its cross-platform compatibility.

TeX is a powerful typesetting system with two main goals: allow anybody to produce high quality documentation and provide a system that would give exactly the same results on all computers. TeX is popular in academia, especially in mathematics, computer science and engineering.

5.4 Example quizzes

I have introduced two cryptography quizzes in order to serve as example.

5.4.1 Message in a column

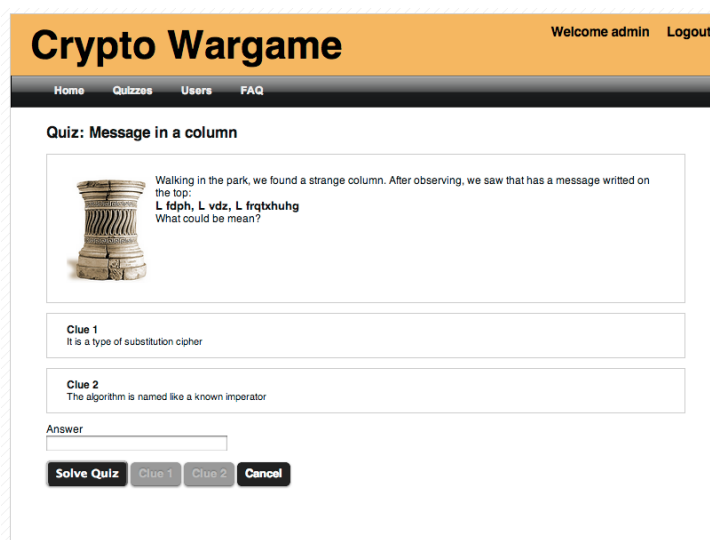


Figure 5.19: Starting level: Message in a column

This could be one of the first quizzes because it is very easy to solve. This quiz is based on the most known encryption technique: The Caesar cipher. The Caesar cipher is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet.

In that case the user only has to recognize the technique and find the replacement length, in order to get the answer.

5.4.2 The cursed book

The cursed book is a quiz when the user has to use more logical skills. The quiz contains an encoded text and a possible password, and the answer is the name of the book.

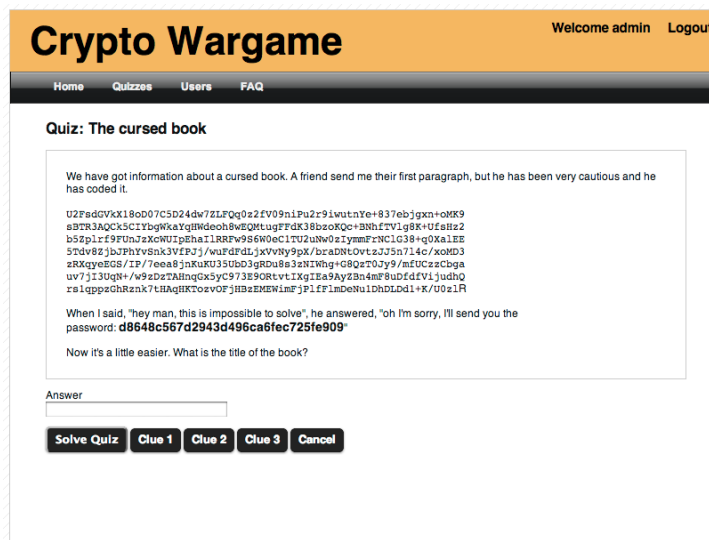


Figure 5.20: Second level: The cursed book

To solve this quiz, first the user have to find that the text is encoded using AES. AES (Advanced Encryption Standard) is a specification for the encryption of electronic data established by the U.S. based on the Rijndael cipher. The algorithm described by AES is a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data.

When the user tries to decode AES with the password provided, it will fail, so he have to try something different. If the user realize that the password is also cyphered, then can try to decode it.

The password is hashed with MD5. A reverse hash will show the real value: *"there is no place"*

When the users apply the decoded password to a AES decoder, the text is revealed:

It was a bright cold day in April, and the clocks were striking thirteen. Winston Smith, his chin nuzzled into his breast in an effort to escape the vile wind, slipped quickly through the glass doors of Victory Mansions, though not quickly enough to prevent a swirl of gritty dust from entering along with him.

So, I he doesn't know the text, only have to google it and discover that the book is: 1984. Then, by answering 1984 the quiz is solved.

Conclusion

6.1 Conclusion

This project has been the creation of a complete wargame platform from zero.

The main goals have been accomplished, and this platform is finished and tested, ready to be used.

When constructing the site I had some security issues in mind:

- SQL Injection
- XSS (Cross-site scripting)
- CSRF (Cross Site Request Forgery)
- Bruteforcing attacks
- Password management (encoded and stored with a salt in the database)
- Validation of the users via email

In general the duration of the tasks were well estimated, with the exception of user management tasks (register, login, recover password,...). Surprisingly, these tasks needed a lot of time, and although not being the most relevant in this project, were necessary in order to get a working platform. Due to that I had less time to improve details or add more security mechanisms.

In relation to the reporting, the fact of using a unknown program (LaTeX), very different from other word processors and writing in a language that is not my first language, turned out to be quite challenging. Even though, both things slowed me down when writing out, it has been an enriching experience.

6.2 Future work

During the development of this work, many interesting research lines have appeared:

6.2.1 More content

The first objective is to fill the platform with more quizzes. Now there are some quizzes with the most popular codes, but the platform has to contain more challenging quizzes.

6.2.2 Dynamic quizzes

This option was proposed at the beginning of this project, but due to the lack of time, we finally dismissed this idea. The objective was that administrators could add some markdowns to the quizzes from the back-end, that could then be calculated from the front-end.

6.2.3 Users can send quizzes

One interesting idea could be that player users could send their own quizzes from the front-end, and then the administrators only would have to validate them in the back-end, and activate them if everything is ok. If a user sends a quizz, the name of the author appears with quiz.

6.2.4 Achievements

In order to improve the playability, it could be interesting to add achievements and get extra points and badges for acomplished tasks.

- Complete X quizzes
- Send a new quiz
- Review a quiz successfully
- Complete X wargames
- More than 1 year registered
- Play in X wargames at the same time
- Send a question that is very hard to solve (many tries, and few successes)
- etc.

Appendices

Installation, Requirements and Deployment

This information also is provided in README file with the code.

A.1 Installation

A.1.1 Database

1. Create a Mysql database
2. Execute in the mysql database the wargame_structure.sql script
3. If you want some example content, you can execute wargame_content.sql (Optional)

A.1.2 Code

1. Copy all the content of the 'public' folder to the site where you want to deploy the wargame
2. Edit config/config.php file with the next information:

DATABASE_HOST : Database hostname. Set 'localhost' value when working at your local. For your webserver, check your hosting provider information

DATABASE_NAME : Name of the database

DATABASE_USER : Username with access to this database

DATABASE_PASS : Password of the username with access to this database

PATH_TEMPLATES : Absolute path to the templates folder

- PATH_TEMPLATES_ADMIN** : Absolute path to the admin templates folder
- PATH_TEMPLATES** : URL where platform is deployed, (Usually www.yourdomain.com)
- EMAIL_INFO** : Email account used to send the register emails
- EMAIL_PASSWORD** : Password of the email account
- EMAIL_PORT** : Port for sending emails (usually 465)
- SMTP_HOST** : Smtplib host (check your email provider)
- PAGE_ROWS** : Pagination of the backend. Number of elements for every page.
- PAGE_ROWS_USERS_FRONTEND** : Pagination of the frontend. Number of users shown for every page.
- SALT** : Salt key used to store users passwords. **WARNING:** Once the app is deployed in production **DO NOT** change this value, or all the user logins will fail.
- DEFAULT_USER** : Default name when a user is registered
- DEFAULT_AVATAR** : Default image name that will be used when a new user is registered. **NOTE:** The image has to be stored in avatar folder

A.2 Software requirements

- Web Server Apache 2.0
- PHP 5.2.4
- PHP PDO active
- Mysql 5.5
- SMTP server

A.3 Deployment

Example program can be found at: wargame.70bits.com

Ramon Fuguet Roma
Barcelona, 2013