Yourjobdone.ie

System documentation

**Document Version: [1.0]**

**Date: [15/12/2015]**

Contents

[Introduction 3](#_Toc437966804)

[Review of possible technologies and solutions 3](#_Toc437966805)

[Security 3](#_Toc437966806)

[Possible language selections 4](#_Toc437966807)

[Forward progressions 4](#_Toc437966808)

[Analysis 5](#_Toc437966809)

[Code documentation 5](#_Toc437966810)

[Implementation 6](#_Toc437966811)

[References 8](#_Toc437966812)

# Introduction

Need for our website i.e. current Irish websites / current market / competitors /

Our project is a find a tradesman based website. Our aim is to make it easier for customers to find a local tradesman with experience and expertise to do the job and make it easier for local tradesmen to find jobs. The tradesmen will create a profile in which they will put up their experiences and hopefully get chosen by the customer based on experiences and perhaps their recent job feedback.

For a large slice of the population, particularly those who have had little or no interest in reaching for the toolbox, the world of domestic skilled trades can be littered with pitfalls, *“is the quote fair?”,* *“Will deadlines be met?”,* sometimes even *“Is the tradesperson qualified?”*, just to name a few. With this in mind, the aim of the team is to develop a fast efficient alternative for the procurement of services across a wide range of skilled trades which will be targeted towards those who have had little or no interest in reaching for the toolbox.

Initial research into the feasibility of the project involved finding out what was already available online of a similar nature. Only three websites were found that were, or could be, comparable to the project outline. Of those three, **findatradesman.ie** was a basic“all-in-one form” style format, which on submission led to a fairly empty database of tradespeople (no plasterers listed in the Dublin area, for example). The other two sites, **tradesmen.ie** and **getworkdone.ie** contained similar functions to the ones this project will include, but the layout and content delivery of the latter was far more polished, providing an easy interface for end users and a comprehensive list of recent work undertaken across all trades with feedback given.

# Review of possible technologies and solutions

## Security

The Open Web Application Security Project (OWASP, 2013) describe XSS attacks as a type of injection in which malicious scripts are injected into otherwise benign and trusted web sites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it, allowing attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites. [2] XSS is listed at number three in OWASP’s 2013 Top Ten threats to web applications. In the same year, WhiteHat identified XSS as its’ joint top threat, citing 53% of their test sample sites as victims of such attacks. [3]

At the top of the OWASP “hit parade”, SQL Injection poses a threat to any application using an SQL database, such as Microsoft SQL Server, Oracle or MySQL, to store data. By modifying standard queries sent to the database, attackers can access entire sets of potentially sensitive data. Described in a recent whitepaper report issued by Cisco, the possible security ramifications range from authentication bypass to information disclosure to enabling the distribution of malicious code to application users. [4]

Again, from the same period, WhiteHat’s statistical analysis conflicts with OWASP’s rating of this threat, claiming that evidence of attacks on web applications by SQL Injection continued its downward slide from 11% in 2011 to 7% in 2012, [5] before falling off the top ten completely by the time of publishing, where it was found to be in 14th place.

Perhaps the most notable aspect of the WhiteHat report was the emergence of other key threats at the top of their table, with Content Spoofing and Information Leakage sharing the Top 3 places with XSS. Accordingly, further research will be undertaken to ensure this and any future projects are safeguarded against these and all relevant threats.

So, what are the best, or indeed the most robust, ways of neutralising the threat posed by these attacks? Beginning with SQL Injection attacks, an article posted on the Enterprise Networking Planet website suggests that from a data input perspective, the most important precautions are data sanitization and validation, which should already be in place. Sanitization usually involves running submitted data through a function (such as MySQL's mysql\_real\_escape\_string() function) to ensure that any dangerous characters (like " ' ") are not passed to a SQL query in data.

On the validation side of the process, basic form protection is often carried out in two ways: by blacklisting dangerous or unwanted characters and/or by whitelisting only those characters that are allowed in a given circumstance, which can involve more work on the part of the programmer. [6]  In fairness, the author does note that client side validation using these methods is not fool proof, so it's essential that you also validate all data on the server side as well.

To follow up on the subject of blacklisting, information was gathered from the W3Schools website, who claim this is not a very good idea. Many of the regularly blacklisted words (like delete or drop) and characters (like semicolons and quotation marks), are used in common language, and should be allowed in many types of input. [7]. In all instances however, every source stresses the importance of including web application firewalls (WAF) and SQL parameters – values such as @ markers, added to the query syntax at execution time.

Looking at the potential threat posed by XSS, all sources advise that to protect against cross-site scripting exploits, enterprises and individuals should make sure they are using the latest version of their browser, to ensure the latest XSS filters are in place. Server administrators should validate input as a matter of course, in addition to carefully restricting the amount of administrator accounts within the network. The OWASP website also includes the XSS Prevention Cheat Sheet, an in-depth guide to protecting web pages from attacks. The cheat sheet contains a list of “rules” to maximise security as well as a summary table detailing samples of each vulnerable attribute and the relative defence method to deploy.

## Possible language selections

## Forward progressions

Forward progression on our website would be by implementing the correct working features which will be a smoother interface and make it more secure for the end user. Since this is the prototype stage of the project, we have the basic design of the website set out with minimum working features. Currently the features working on our website is when the user registers, they will receive a “confirmation” email regarding their sign up. So far the user just receives an email containing their name, date of birth, job, etc… We plan on implementing the randomizer for confirmation codes, which the user will need to enter manually or by clicking the link in the email which is generated by our system.

We plan to implement other simple features such as phpBB which is a free and open source forum software for the customers to post ads for jobs they need done. With this feature set in place, it will make it easier for customers to post quick ads for jobs. Tradesmen who wish to take this job upon themselves may contact the customer who posted the ad. We have two options of implementing this feature on our website, one is to download the software from their website and install it ourselves, two would be to change hosting to a hosting service which provides this feature integrated already with easy install.

Password recovery is another feature which needs to be implemented into our website. With users forgetting their passwords on a regular basis, we will implement password recovery service in which the user will enter their username or email in which they will receive an email containing a generated link to reset the password of that account. We can do this in two ways, one is send an email reset link, two would be asking the user security questions which they answered during registration to reset their password.

Since our project requires people to register and login. We plan to get a SSL certificate for security reasons. In the future we might plan to introduce featured tradesmen, in which the tradesman could purchase this feature for their profile. This feature could last 7 days to a month, it depends on how we decide in the future. The tradesman profile will become featured (highlighted) when customers search for tradesmen.

# Analysis

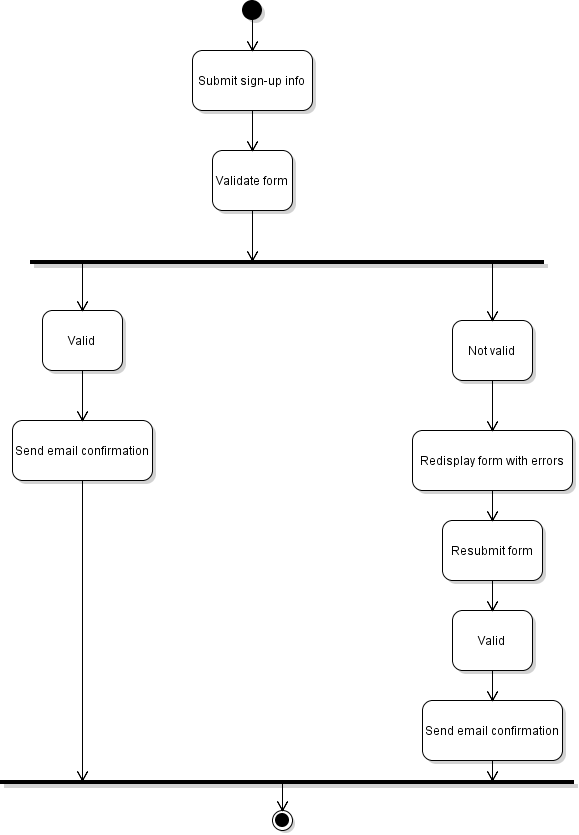
## Code documentation

# Implementation

User Login activity diagram:



Sign up activity diagram:



Password recovery:

Posting job:

# References

[1], [3], [5]. WhiteHat Security. 2013. *Website Security Statistics Report*. [ONLINE] Available at: <https://www.whitehatsec.com/assets/WPstatsReport_052013.pdf>. [Accessed 20 November 15].

[2]. Open Web Application Security Project. 2013. *Top 10 2013*. [ONLINE] Available at: <https://www.owasp.org/index.php/Top_10_2013-Top_10>. [Accessed 20 November 15].

[4]. Cisco Systems. 2012. *Understanding SQL Injection*. [ONLINE] Available at: <http://www.cisco.com/web/about/security/intelligence/sql_injection.html>. [Accessed 19 November 15].

[6]. Paul Rubens. 2010. *10 Ways to Prevent or Mitigate SQL Injection Attacks*. [ONLINE] Available at:<http://www.enterprisenetworkingplanet.com/netsecur/article.php/3866756/10-Ways-to-Prevent-or-Mitigate-SQL-Injection-Attacks.htm>. [Accessed 20 November 15].