



**Technical Solution Design**

**for**

**Budding Sharemarket Investor**



Version: V1.0

Date: 18/09/2016

Sponsor: RMIT

Author: Carlo R Beasley, Lucas Brook, Evan Le Clercq, Vio Marcu, Ocal Ogten

Commercial - in – Confidence

Document Control

**Distribution**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Issued** | **Recipient** | **Position** |
| V 1.0 | To be submitted when required and the date recorded here | Amir Homayoon Ashrafzadeh | Supervisor |
|  |  |  |  |

**Amendment History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Section** | **Page** | **Version** | **Comment** |
| <Enter Doc. Section No.> | <Enter Page No.> | <Enter Version No.> | <Enter Comments to explain the reason for the document text or other changes,    e.g., Updated text after walkthrough with the stakeholders, or    e.g., Updated section after technical consultation> |
|  |  |  |  |

.

**Staff or Entities Consulted**

|  |  |
| --- | --- |
| **Name** | **Position / Organization** |
| Carlo R Beasley  Lucas Brook  Evan Le Clercq  Vio Marcu  Ocal Ogten | Development Team  Project manager  Development Team  Project Manager  Development Team |

**Related Documents**

|  |  |  |
| --- | --- | --- |
| **Name** | **Author** | **Description** |
| <Enter Document Name> | <Enter Author> | <Enter Document Description> |
|  |  |  |

***Preface***

The purpose of this document is to outline the Technical Solution Design for Budding Sharemarket Investor application project.This document describes the technical environment, overall project architecture, system architecture, application functionalities and features, database architecture, implementation instructions, non-functional specifications, summary of test results, known issues and risks.

**Table of Contents**

**1 Introduction 1**

**2 Technical Environment 1**

**3 Overall Architecture 1**

**4 System Architecture 1**

**4.1 Functionalities/features 1**

**4.1.1 Functionality 1 1**

**4.1.2 Functionality 2 1**

**4.1.3 Functionality N 1**

**5 Database Architecture 1**

**6 Implementation Instructions 1**

**7 Non-functional specifications 1**

**8 Summary of test results 1**

**9 Known Issues & Risks 1**

**10 Other Considerations 2**

**11 Appendix 2**

# 

# 

# **1** **Introduction**

Summary of the technical solution that was completed including:

· Brief project description

· Brief description of technical environment

· Estimated level of complexity

· **Estimated benefits/Problems it solved (quantitative & qualitative)***.*

The Budding Sharemarket Investor game will allow players to create an account, log into the web application, use bogus money to simulate buying and selling of shares based on actual ASX data. Data will be retrieved via web service calls to a finance API, and will be used in the buying, selling and tracking of shares in the market.

The budding investor game will be hosted online AWS and built with a combination of PHP, mySQL, HTML and CSS. Users will be able to navigate to the site, login or register, and begin dealing in the faux marketplace. There will also be admin functionality built into the site. GitHub will be used for a version control tool, and some testing tools such as Unit Testing and PHP Testing.

The application will initially display a lower level of complexity and as the project development progresses, it will transition into a moderate level of complexity while remaining objective to the project requirements.

**Estimated benefits/Problems it solved (quantitative & qualitative)***.*

# **2** **Technical Environment**

<Detail the technical environment/technologies used to complete this project and reason for the choice of the specific technology that is used. For instance, if PHP is used, why it is chosen and if GITHUB etc is used for source control please specify those details as well..>

**PHP** - is the main language chosen for a few reasons such as, being server-side and not client-side therefore it is more secure and has the ability to do database queries and a more secure login.

HTTP, CSS, AJAX,

**GitHub** - is used for hosting our source control, as it integrates with other software used such as Source Tree and can also have more than 5 users connected to it at no charge unlike BitBucket which is limited to 5 users.

**Source Tree** - is used to ease the project works offering tasks such pull, commit, push, merge, create other branches, whilst connected to the GitHub source control

**Sublime Text, Notepad++** - tools of choice, used individually by group members to perform coding

**Laravel** - our main framework used to build the PHP project, structuring the location of pages, styling, scripts and databases

**AWS** - product host, it is where we keep track of different versions and where client can interact with the product: http://clevo-rmit.space/public

**Lean Testing** - platform used to test our project functions

**Trello** - used to manage the project sprints - current, upcoming, done, and ongoing

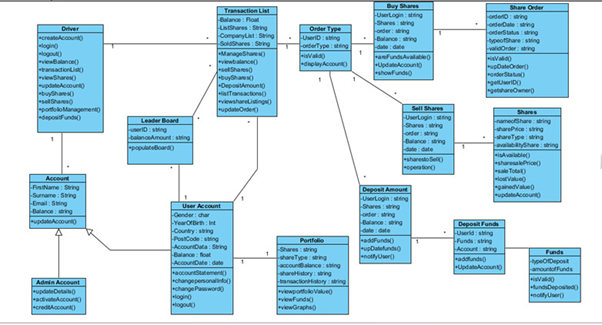
**Slack** - tool used for group communication, structured into multiple channels such as API, UI, General…

**XAMPP** - used locally to run the project individually before pushing to GitHub

**Google Drive** - where we have the main copy of our documentation but also replicated within the source code in GitHub

**3** **Overall Architecture**

<Outline the overall architecture of the solution which details how the system will interact with the world or other systems etc. Explain it using a diagram.>



# **4** **System Architecture**

<Detail the system that was built/completed. Explain each component thoroughly. A architecture diagram is essential. >

## **4.1** **Functionalities/features**

<Detail the individual/specific functionalities that comprise the system.>

1. User Account - Sign Up
2. User Login / Logout
3. User buying and selling of shares
4. Applications provides live updates of share prices
5. Application provides a history of share price fluctuations
6. Application provides player’s current balance
7. Application maintains an updated leader board
8. Application offers appropriate admin functionality

### **4.1.1** **Functionality 1**

<explain each functionality using a flow chart>

### **4.1.2** **Functionality 2**

### **4.1.3.1 BUY SHARES**

1. User logs in
2. User navigates to Transactions page
3. User will select desired stock from the ‘Search Live Stock’ box.
   1. This will update the buy boxes with company information
   2. The buy box will update with number of shares currently owned of the company and the number maximum number of shares that can be bought with the user’s current balance.
4. User will select number of the shares that will be purchased.
5. Total cost of shares to be updated.
6. User will push the ‘Buy Shares’ button
7. Application to check if the user has sufficient funds to make transaction.
   1. If insufficient funds error message will show.
   2. If sufficient funds will proceed to step 7.
8. Application to calculate the total cost of the transaction and update the ‘current\_balance’ field in the database.
9. Application will update the current\_holdings field in the database with the new stocks. This will be hashed to increase security of the information.
10. The user will be notified of the success of the purchase and the amount taken from their balance.

**4.1.3.2 SELL SHARES**

1. User logs in
2. User navigates to Transaction page
3. User will select desired stock from the Owned Shares’ dropdown menu.
   1. This will update the sell box with company information
   2. The buy box will update with number of shares currently owned of the company.
4. User will enter the number of shares to be sold.
5. Total value of shares to be sold updated.
6. User will push the ‘Sell Shares’ button
7. Application to confirm the number of shares entered is equal to or less than the number currently owned.
   1. If too many shares entered error message will be displayed
   2. If number of shares are being sold is valid will proceed to step 7.
8. Application to calculate the amount to be added to the user balance and update the ‘current\_balance’ field in the database.
9. Application to update the ‘current\_holdings’ field in the database with the new stock information.
10. User will be notified of the success of the sale and amount added to their balance.

# **5** **Database Architecture**

<Explain the database architecture/schema and why it is built in the way it is and how scalable it will be.>

# **6** **Implementation Instructions**

< List the implementation instructions and the basic specs of the server needed. Also provide details regarding data migration or if its replacing or enhancing an existing system what are the steps needs to take to ensure there is no data loss.>

# **7** **Non-functional specifications**

<Detail all the non-functional specification of the system.>

# **8** **Summary of test results**

<Provide the summary of the test cases and results in tabular format.>

# **9** **Known Issues & Risks**

<Outline any known issues and/or risks that are likely to impact or be caused by this initiative in any way.>

# **10** **Other Considerations**

<Discuss any other considerations for this project proposal’s acceptance and delivery.>

# **11** **Appendix**

<Refer the tool that is used to capture the functional requirement and if possible provide the references to the tool and also a summary of the functional requirement. It can be simply compilation or copy of the user stories from JIRA/Rally.>

<NOTE: These headings are guidelines only. Based on your project, you might require additional headings, so feel free to add headings as required.>