

Software Engineering Group Project Design Documentation

Authors: Ben Dudley, David Fairbrother, Jonathan Englund,
Josh Doyle, Liam Fitzgerald, Maurice Corriette,
Oliver Earl, Tim Anderson
Config Ref: SE_05_DEL_03
Date: 2015-11-26
Version: 1.0
Status: Release

CONTENTS

CONTENTS	2
INTRODUCTION	3
1.1 Purpose of this Document	3
1.2 Scope	3
1.3 Objectives	3
2. DEPLOYMENT DESCRIPTION	3
2.1 Applications in the system	3
2.2 Application interactions	4
3. INTERACTION DESIGN	5
3.1 Use-Case Diagrams	5
3.2 User Interface Design – Tasker CLI	6
3.3 User Interface Design - TaskerMAN	11
4. COMPONENT DESCRIPTION	19
4.1 TaskerSRV Database Design	19
5. SIGNIFICANT CLASSES	20
5.1 TaskerCLI	20
5.2 TaskerMAN	23
6. DETAILED DESIGN	24
6.1 Activity Diagrams	24
6.2 Sequence Diagrams	28
6.3 TaskerCLI Data Structures	33
6.4 Spike Work	35
DOCUMENT HISTORY	36
APPENDICES	37

INTRODUCTION

1.1 Purpose of this Document

The purpose of this document is to describe and specify a full design for all software within this project. This will be used by software engineers to implement several components. It also lists the integration of these components to facilitate a solution matching the client's requirements [1].

1.2 Scope

This document shows a complete and full design for TaskerCLI, TaskerMAN and TaskerSRV. It lists the requirements to run each component and how they integrate and communicate to each other. It goes on to list how these components will be implemented specifying in detail algorithms where required.

1.3 Objectives

This document contains a complete view of the software solution designed. Initially the design considers high level aspects of design such as typical use cases and GUI mock ups it. It then lists classes used within these various components which are further broken down in complex classes into activity diagrams, sequence diagrams and spike work conducted. The reader should be able to visualise the overall look of the software whilst understanding the implementation at lower levels of design.

2. DEPLOYMENT DESCRIPTION

2.1 Applications in the system

2.1.1 TaskerCLI

TaskerCLI is the desktop based application in the system. The software will be written in Java and will be tested with Java 1.7.0_85 running on a Linux 64-bit Operating System. [Appendix A] - using versions of the Java Runtime Environment lower than this may cause unexpected behaviour and therefore is not recommended.

JDBC will be used to facilitate data communication. The version this software will be developed with is 4.2, utilising driver version 5.1.37.

The JUnit testing framework that is used during development will be version 4.12. This requires Java Development Kit 1.5 or above. [1]

2.1.2 TaskerMAN

TaskerMAN is the web-based software component of the system. The website will be built with HTML5, CSS (Cascading Style Sheets), JavaScript and PHP. The PHP tested during development is PHP Version 5.6.13 [Appendix B] running on an Apache server [Appendix C], running on Gentoo Linux 3.18.7 64-bit [Appendix D].

This information is also available by running *phpinfo()* on the targeted web server. [2]

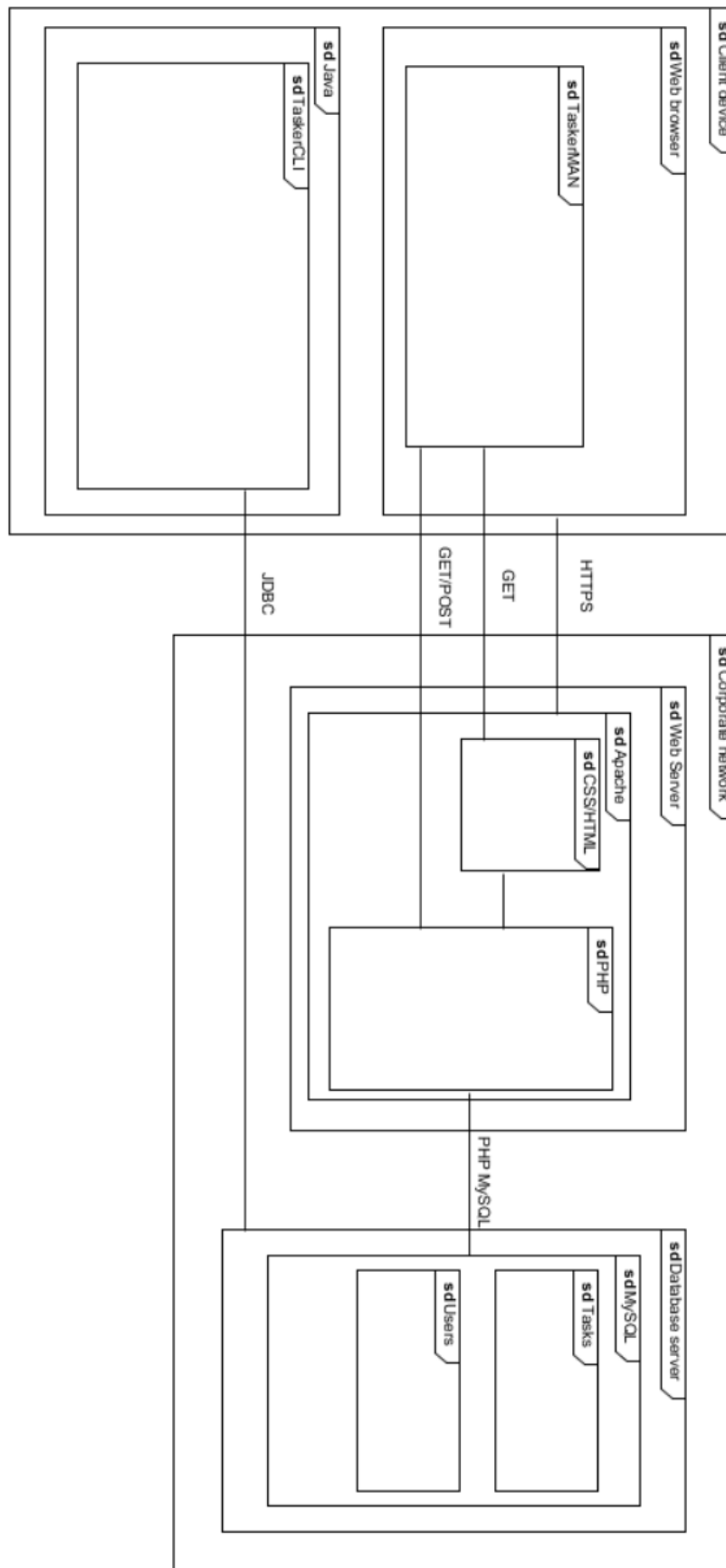
In order to enable the use of the PHPUnit testing framework, a minimal installation of PHP 5.6 is required, but the latest install is highly recommended. [3]

2.1.3 TaskerSRV

TaskerSRV is the database component.

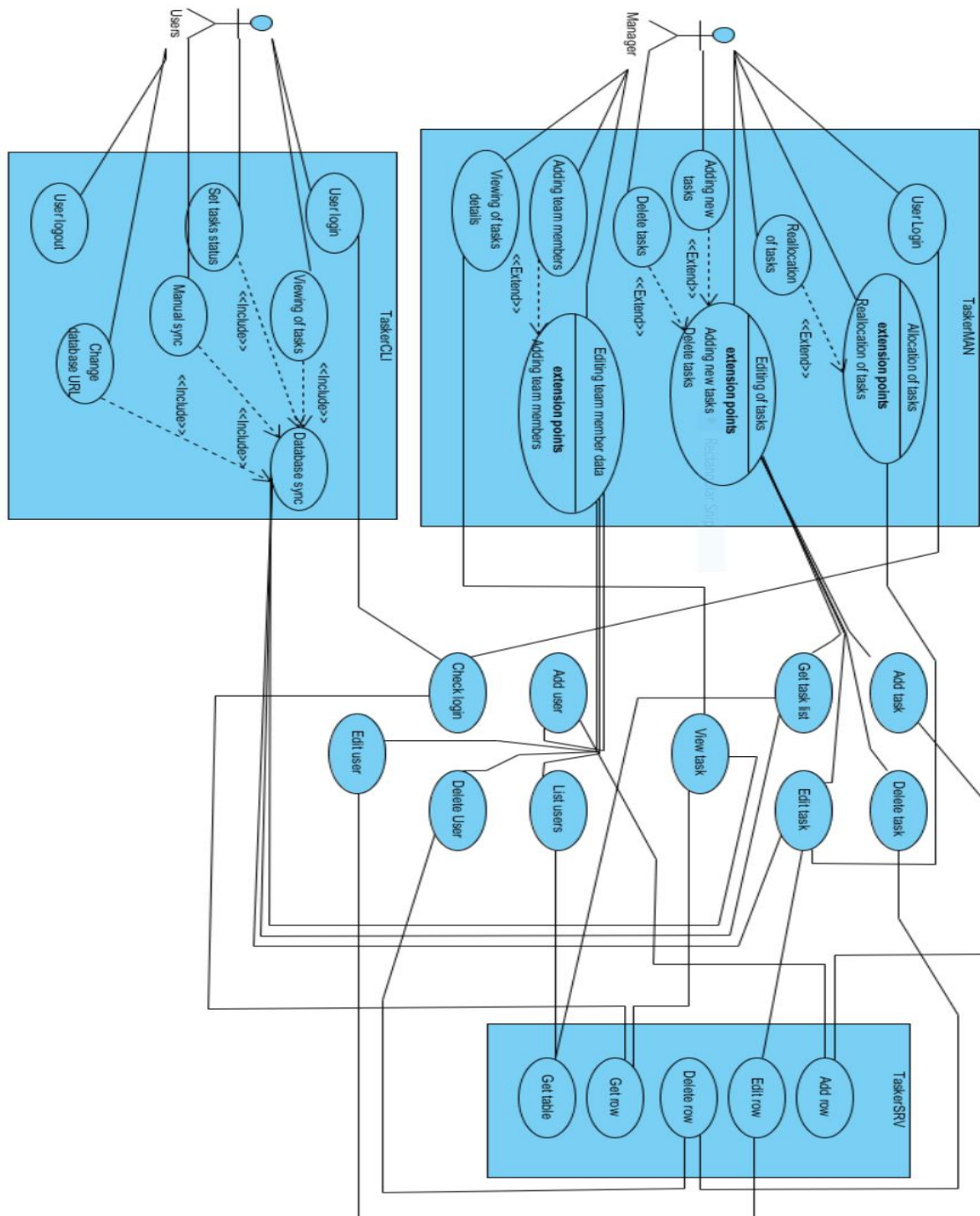
A MySQL relational database will be used. The version will be tested against is MySQL 5.6.26 on a Linux 64-bit Operating System [Appendix E]. The main system requirement for a current MySQL installation is 2.5GB of free hard disk space [4], and any disk space pertinent to the size of the database.

2.2 Application interactions



3. INTERACTION DESIGN

3.1 Use-Case Diagrams



3.2 User Interface Design – Tasker CLI

3.2.1 Log In Window

The image shows a graphical user interface for a window titled 'TaskerCLI'. The window has a title bar with the text 'TaskerCLI' and three standard window control buttons (minimize, maximize, close) on the right. The main content area is light gray and contains the following elements: a label 'Email:' followed by a white rectangular text input field; a light gray button labeled 'Log In'; and another light gray button labeled 'Connection Settings' positioned below the 'Log In' button.

- Textbox for user to enter their email address for validation, as per requirements specification. [6]
- Log In button opens 'Main Window' when clicked, provided a valid email address has been entered.
- Clicking the 'Connection Settings' button will open the 'Connection Settings' window – this allows the user to configure their connection to the *TaskerSRV* database.
- Closing 'Log In Window' will bring up the 'Exit Confirmation' window.

3.2.2 Main Window

TaskerCLI

Quick View (1/2)

Task Name: TaskerCLI UI Design

Status: Allocated

Assigned Task Member(s): Josh Doyle

Start Date: 21/10/2015

Expected End Date: 25/10/2015

Task Elements: Draw design for login window, draw design for main window, draw design for edit window, draw design for delete confirmation window, write comments, add any references if needed

< Edit >

Connection Status

Online

Last Synced 0 Minutes Ago

Connection Settings

	Task Name	Status	Assigned Task Member(s)	Start Date	▼ Expected End Date ▲
<input checked="" type="checkbox"/>	TaskerCLI UI Des...	Allocated	Josh Doyle	21/10/2015	25/10/2015
<input checked="" type="checkbox"/>	Project Plan	Allocated	David Fairbrother	17/10/2015	27/10/2015
<input type="checkbox"/>	Test Specificat...	Allocated	Maurice Corriette	20/10/2015	09/11/2015

- The table at the bottom of the 'Main' window shows all of the tasks currently saved in *TaskerSRV*, the last time the program was synchronised with the database.
 - Clicking on the headings at the top of the columns in the table, will order the table based upon the values in that column. The design shown is in descending order based upon the *Expected End Date* column.
 - Checking the checkboxes next to each task, enables the user to select multiple tasks.
 - The scrollbars are used to navigate the table.
 - Selected tasks are shown in more detail in the 'Quick View' panel.
- The Quick View panel at the top left of the 'Main' window presents the data from the tasks selected from the table.
 - Clicking the arrow keys at the bottom of the Quick View panel navigates between all tasks selected from the table at the bottom of the 'Main' window.
 - Clicking the 'Edit' button opens the 'Edit' window, to change completion status and task elements of the current task in the 'Quick View' panel.
- The Connection Status panel at the top right of the 'Main' window status changes colour depending on the connection status.
 - Green indicates that *TaskerCLI* is currently connected to *TaskerSRV* and that everything is synchronised.

- Red indicates that the connection between *TaskerCLI* and *TaskerSRV* has been lost and that synchronisation is no longer guaranteed.
- In the demonstrated design, *TaskerCLI* is connected to *TaskerSRV* and has been synchronised less than a minute ago. The number of minutes increments every minute and returns to 0 after successful synchronisation.
- Closing the window brings up the 'Exit Confirmation' window.

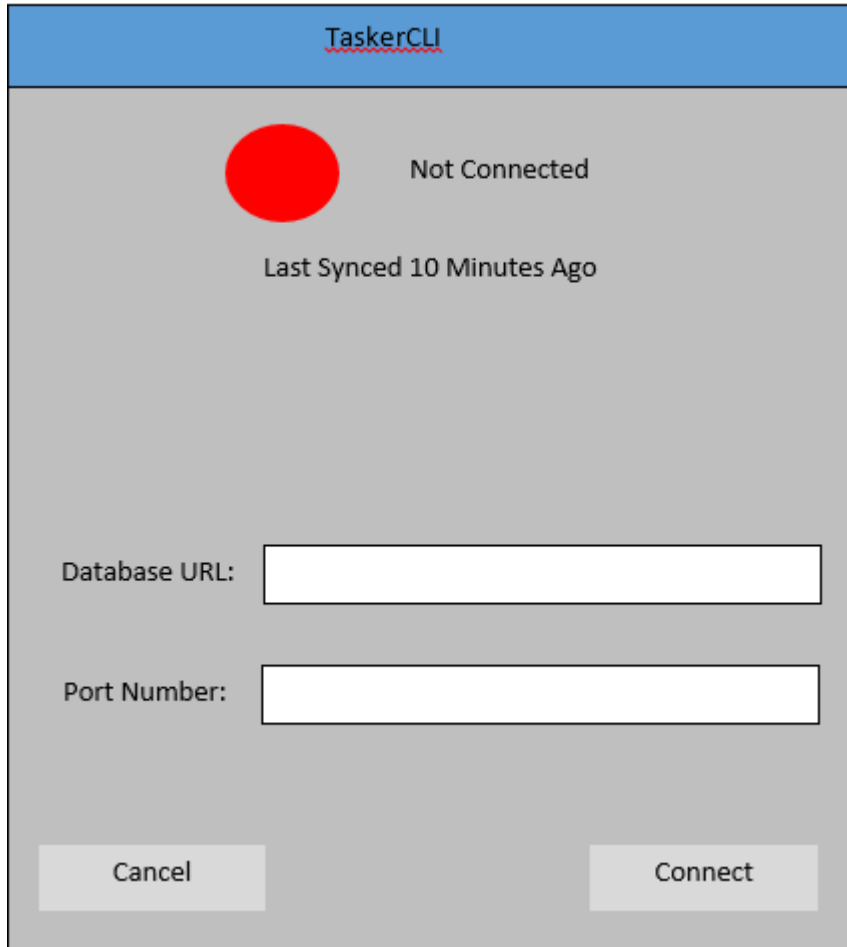
3.2.3 Edit Window

The screenshot shows a window titled 'TaskerCLI' with a subtitle 'Edit Task'. The window contains the following fields and controls:

- Task Name:** TaskerCLI UI Design
- Status:** Assigned (with a dropdown arrow)
- Assigned Task Member(s):** Josh Doyle
- Start Date:** 21/10/2015
- Expected End Date:** 25/10/2015
- Task Elements:** A text area with a scrollbar containing the text: 'Draw design for login window, draw design for main window, draw design for edit window, draw design for delete confirmation window, write comments, add any'.
- Buttons:** Cancel and Save.

- The 'Edit Task' window is populated with the data of the task that was in the 'Quick View' panel on the 'Main' window when it was opened.
- The attributes of completion status and task elements are editable from this window.
- The completion status can be selected from a dropdown list. The default value is 'Assigned.'
- The task elements can be changed by typing into the Task Elements textbox.
 - A scrollbar will only appear if the text entry exceeds the size of the textbox.
- When the Save button is clicked, the 'Edit Task' window is closed and the task attributes are updated with their new values.
- Choosing Cancel simply closes the window with no changes.

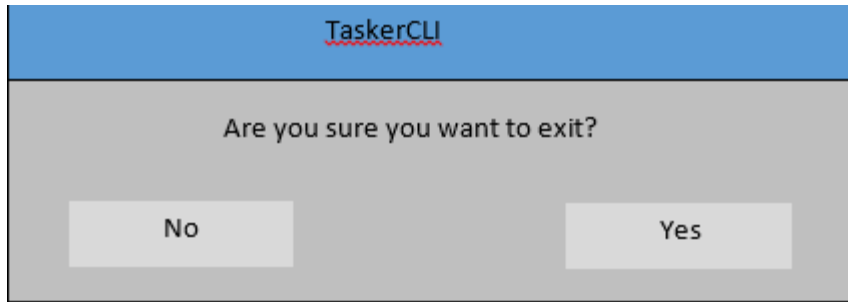
3.2.4 Connection Settings Window



The image shows a window titled "TaskerCLI" with a blue header bar. The main area has a grey background. At the top, there is a red circle and the text "Not Connected". Below this, it says "Last Synced 10 Minutes Ago". There are two input fields: "Database URL:" and "Port Number:". At the bottom, there are two buttons: "Cancel" and "Connect".

- The connection status text at the top of the window shows the current connection state of *TaskerCLI*.
 - The coloured circle is red when there is no connection established to *TaskerSRV*.
 - Consequently the coloured circle appears green when a connection is successfully established.
 - The time since last sync shows how much time has passed since the last synchronisation.
 - In this design, *TaskerCLI* is not connected to *TaskerSRV* and it has been 10 minutes since the last successful synchronisation.
- The Database URL and Port Number are entered into the respective fields to provide information for connecting to the *TaskerSRV* database.
- Choosing 'Cancel' simply closes the window without saving any information.
- Choosing 'Connect' will instruct *TaskerCLI* to attempt to connect using the information provided.
- Default window controls and clicking outside of the window are disabled to prevent the user from opening multiple instances of this window and attempting to cause simultaneous connections to be established.

3.2.5 Exit Confirmation Window



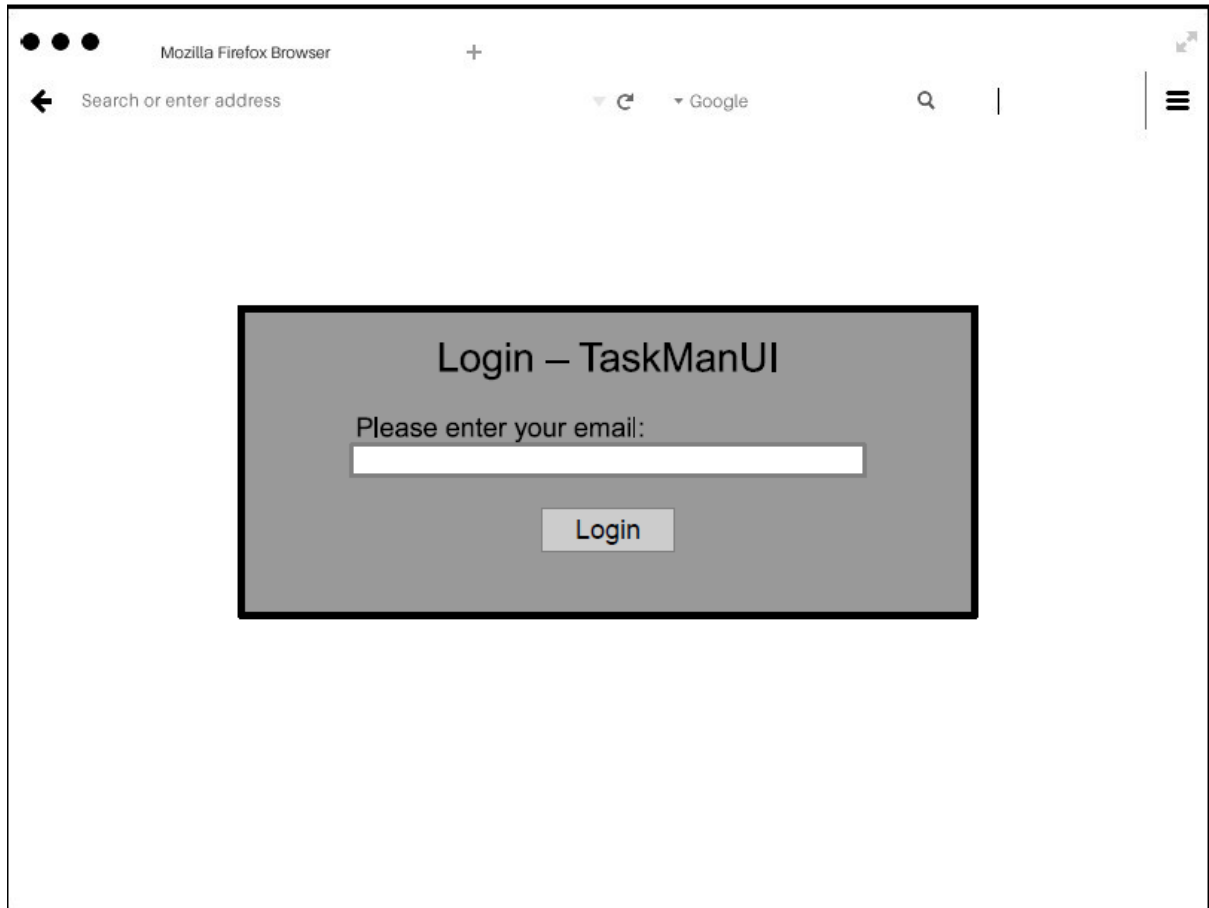
- If 'No' is selected, the window is closed and the user regains control of the window they were previously using.
- If 'Yes' is selected, *TaskerCLI* closes.
- Default window controls are disabled to make it clear to the user that their attention is required and that a decision must be made.
- Clicking away from the window to bring another window into focus is also disabled, to stop the user spawning multiple instances of the 'Exit Confirmation' window.

3.3 User Interface Design - TaskerMAN

3.3.1 General Notes

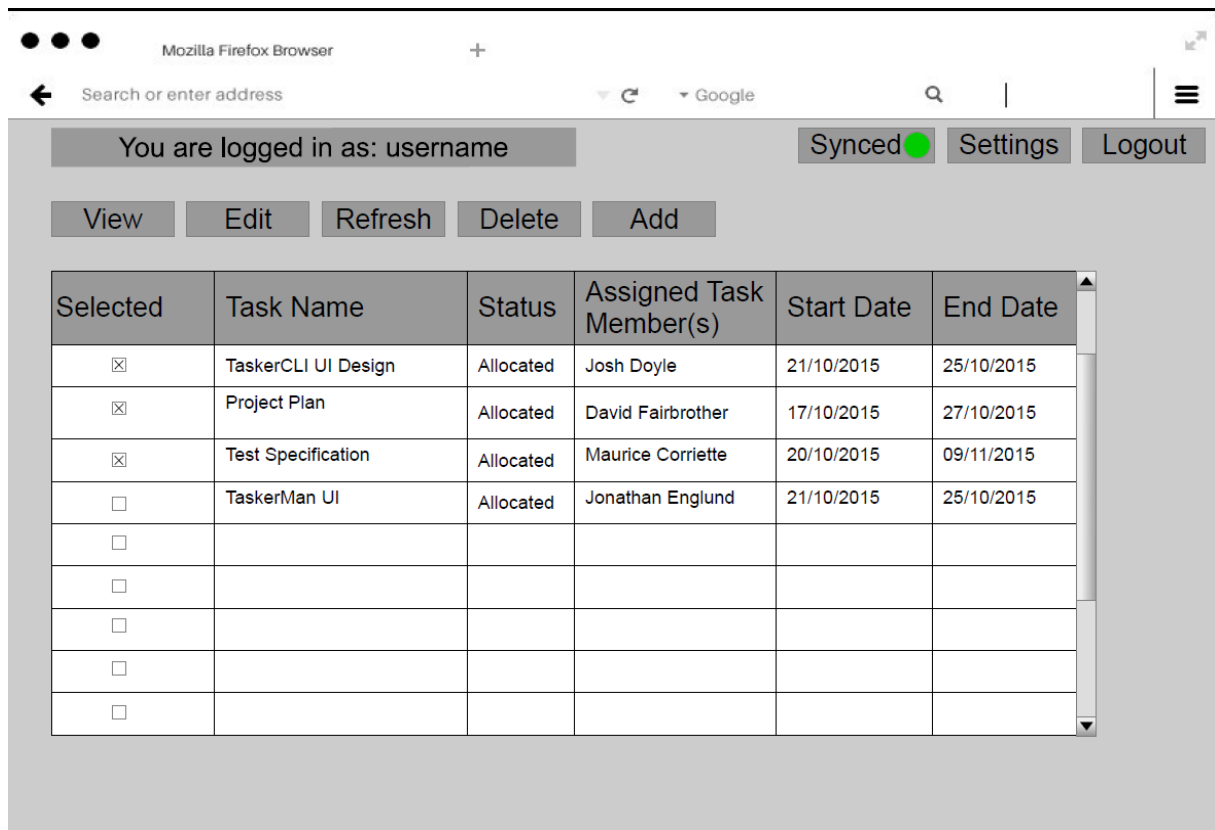
Mozilla Firefox is used as an example web browser in the following images. [6]

3.3.2 Login Page



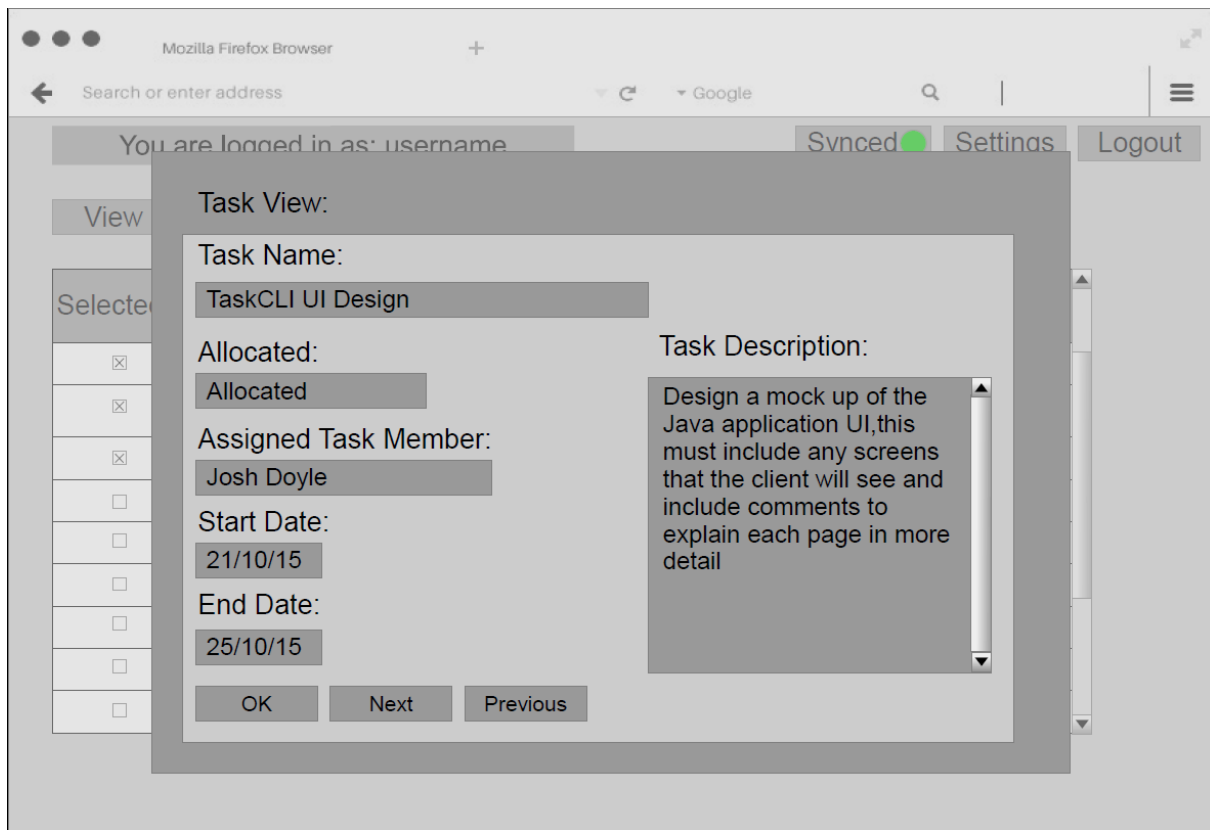
- If a valid email address is entered, the user will be directed to the main page – otherwise access is prohibited, as is required. [6]

3.3.3 Main Page



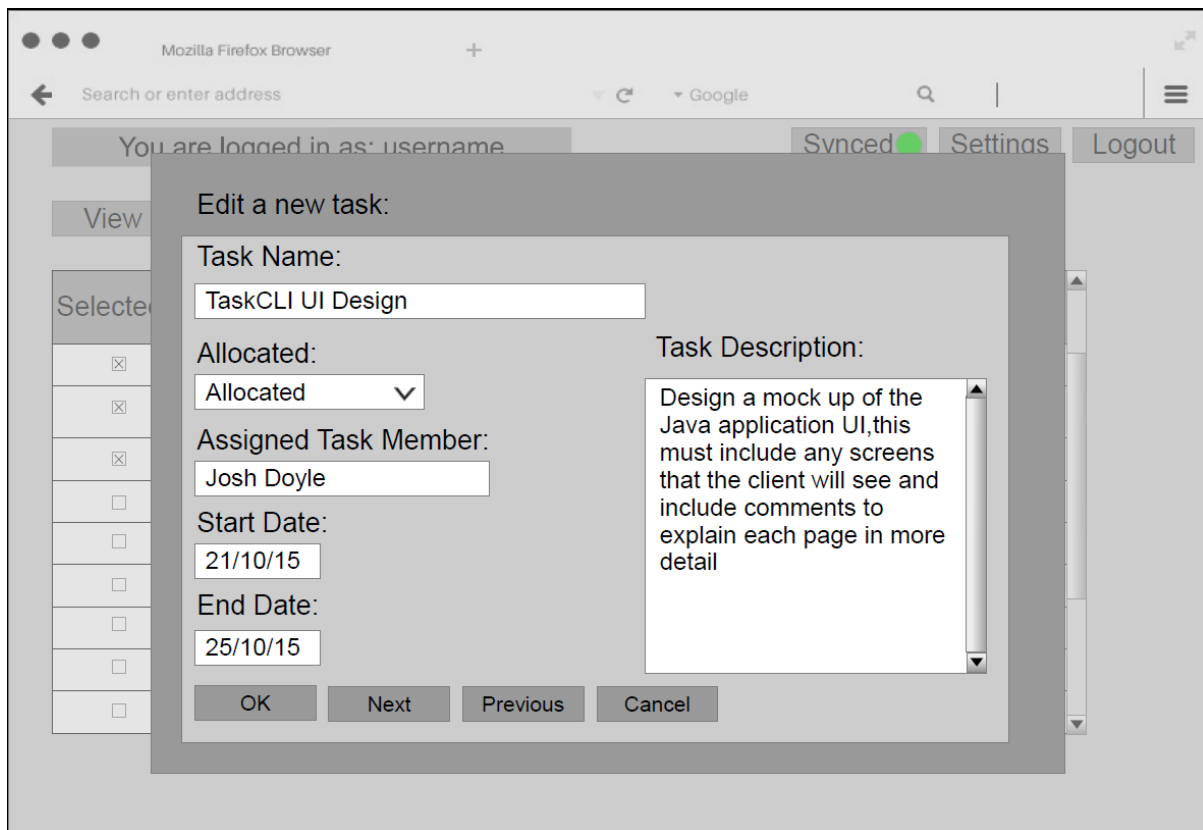
- Displays user's email address at the top of the page, indicating who is currently logged in.
- The 'Logout' button displays the 'Log Out Prompt', and directs the user to the 'Logout Successful' screen upon successful logging out.
- The 'Synced' button will show a green dot if currently synchronised with *TaskerSRV*. Red if not.
- The 'Settings' button will bring up the 'Connection Setting' screen.
- All information shown in the database table and tasks can be selected and edited in bulk.
- Buttons at the top provide functionality for viewing, adding, editing and deleting of tasks, or refreshing the table.

3.3.4 View Task Overlay



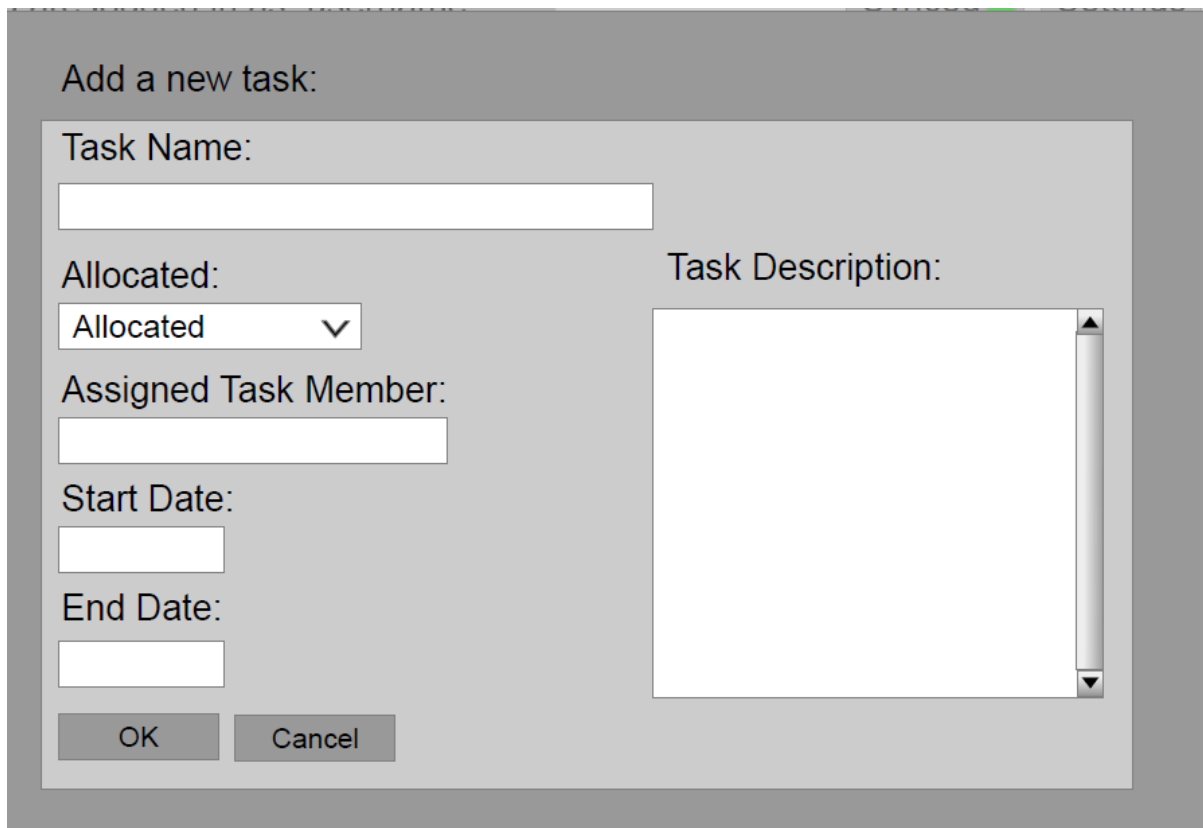
- Displays selected task. 'Next' and 'Previous' allow the browsing of other entries.
- Read-only

3.3.5 Edit Task Overlay



- By selecting one or more tasks, the 'Edit Task' overlay appears where details can be changed. Clicking 'OK' will save these changes. 'Next' and 'Previous' allow the navigation through other entries.
 - Next/Previous do not appear if the user has only selected one task.
- The database will be updated after each edit, so one task can be modified before cancelling.

3.3.6 Add Task Overlay



The image shows a software overlay window titled "Add a new task:". The window has a light gray background and a darker gray border. Inside, there are several input fields and a large text area. The "Task Name:" field is a single-line text box. The "Allocated:" field is a dropdown menu with "Allocated" selected. The "Assigned Task Member:" field is a single-line text box. The "Start Date:" and "End Date:" fields are single-line text boxes. The "Task Description:" field is a large multi-line text area with a vertical scrollbar. At the bottom, there are two buttons: "OK" and "Cancel".

Add a new task:

Task Name:

Allocated:

Assigned Task Member:

Start Date:

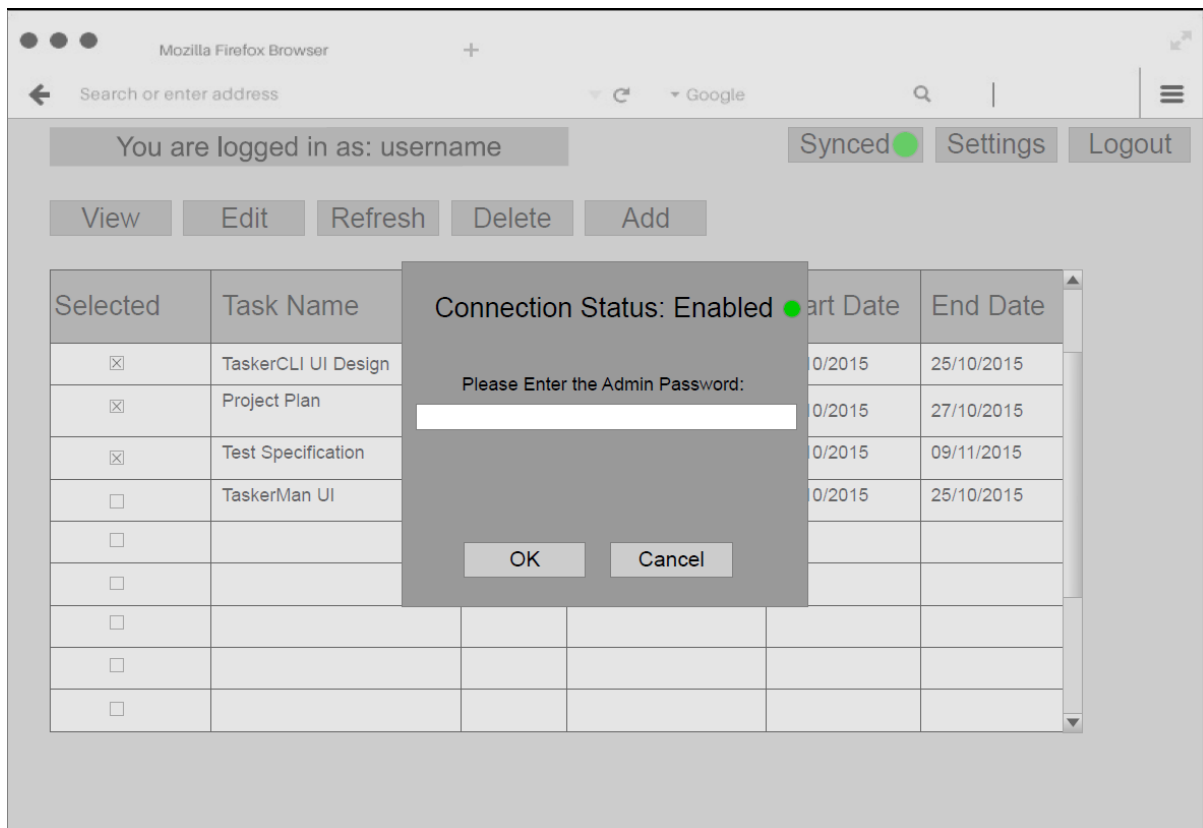
End Date:

Task Description:

OK Cancel

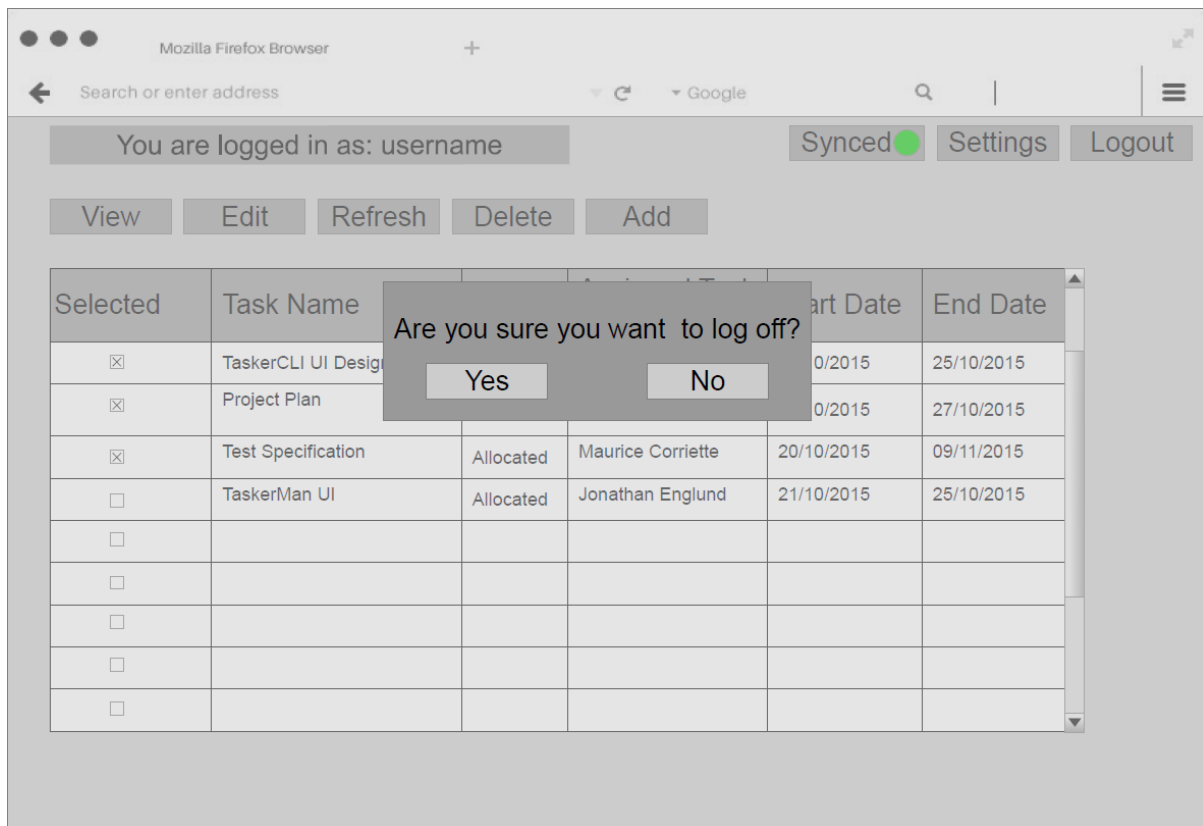
- In visual likeness to the 'View Task' screen, except blank where information can be inserted.
- Validation will be used to ensure only correct/meaningful data can be entered.

3.3.7 Connection Setting Screen



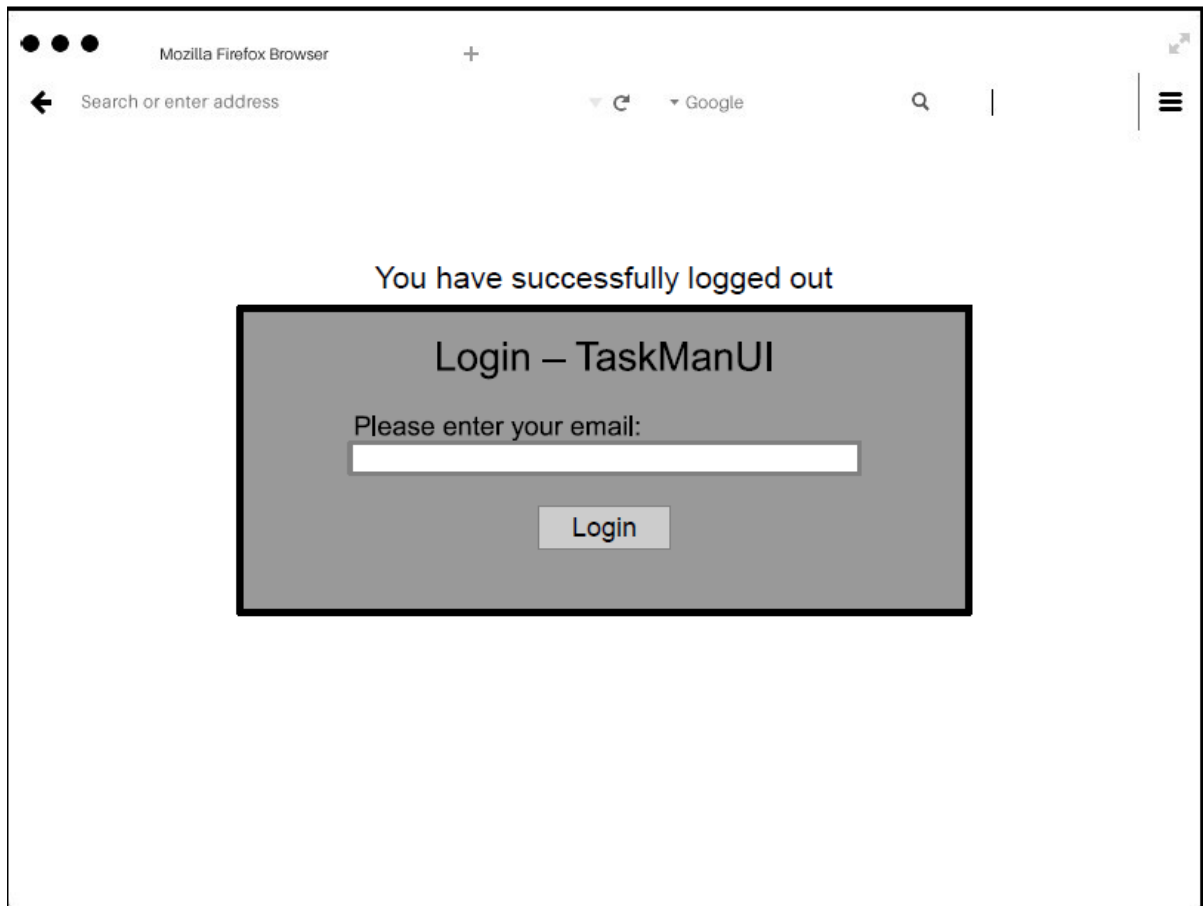
- Shows current synchronisation/connection status to *TaskerSRV*.
- Connection settings can be modified, but as this is a web based client, it is hidden behind an administration password wall and can be only modified by administrators.

3.3.8 Log Out Prompt



- Prompts the user whether or not they wish to logoff. Choosing 'No' returns the user to the 'Main Page' screen. Choosing 'Yes' directs the user to the 'Logout Successful' screen.
- The user cannot dismiss this prompt by clicking outside of its boundaries.

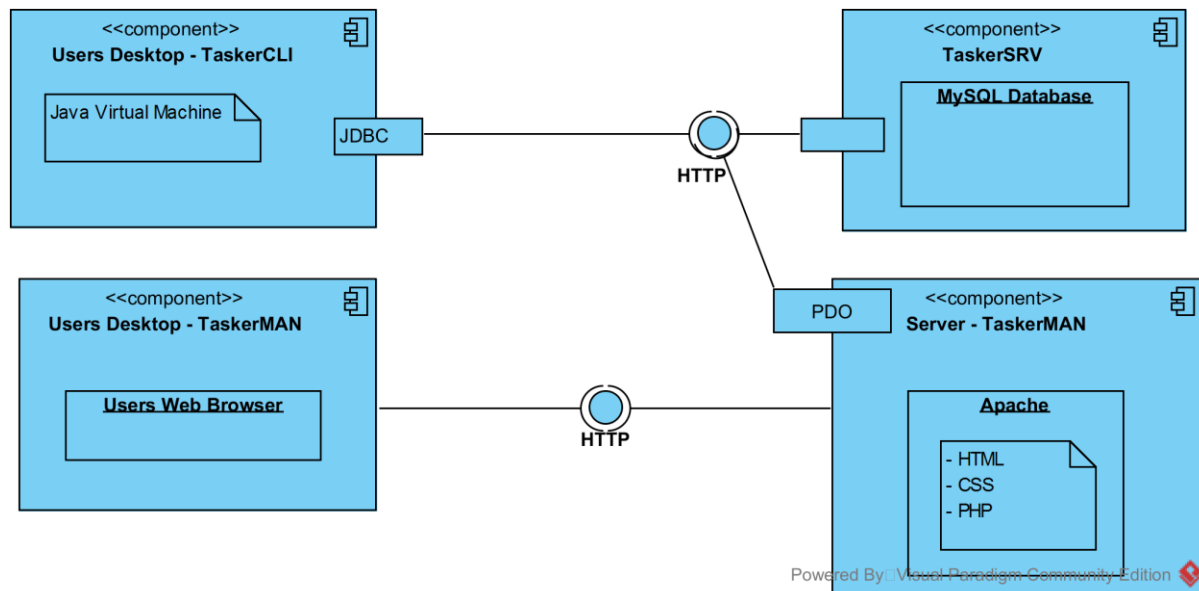
3.3.9 Logout Successful



- User has been successfully logged out and is informed that they must login again to use the system.

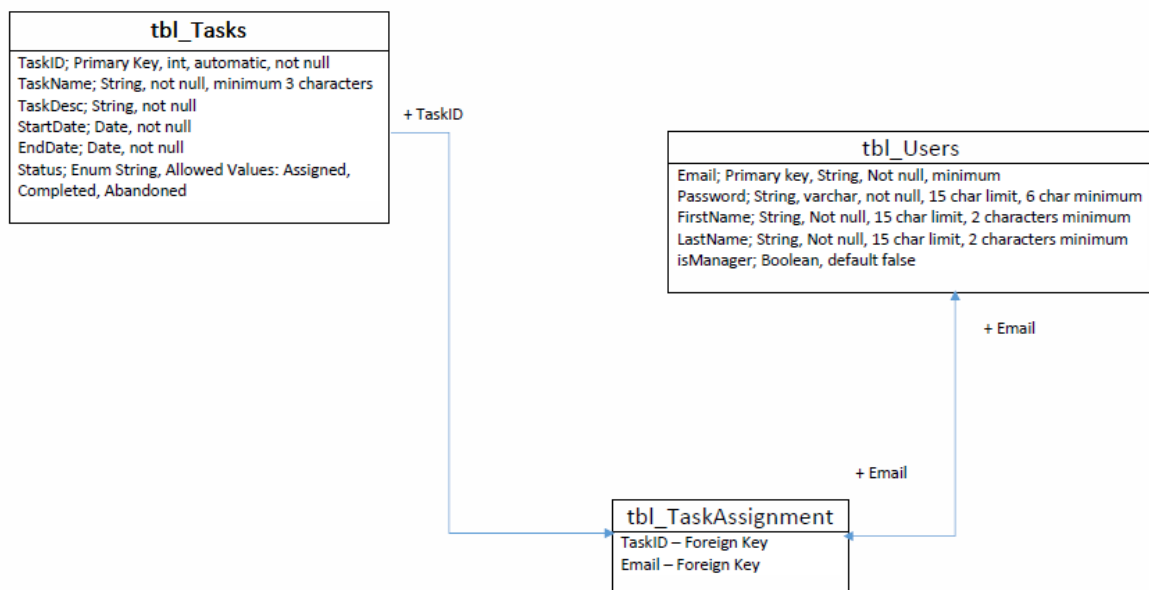
4. COMPONENT DESCRIPTION

By using standard libraries to connect various components to TaskerSRV we can utilise standard protocols such as HTTP avoiding the need to specify interfaces for inter component communication. Clients will initiate the connection to TaskerSRV and handle the connection through JDBC and PDO for TaskerCLI and TaskerMAN respectively.



4.1 TaskerSRV Database Design

Using this design the database in TaskerSRV must use a standard naming scheme and have fixed properties. These are listed in the diagram below.



5. SIGNIFICANT CLASSES

5.1 TaskerCLI

TaskerCLI classes can be broken down into functional groups. Classes which handle data including editing, database synchronisation and ordering are grouped as “Logic Classes. The remaining Classes are used to power the GUI such as getting user inputs and displaying or closing windows, these are grouped as “GUI Classes”

5.1.1 Logic Class Diagram

See Appendix F – Logic Class Diagram.

The classes and descriptions are as follows:

Database: - Holds a JDBC connection and performs execution of SQL statements in order to both send and receive data to TaskerSRV.

Task: - Represents an individual task and holds the name, elements, dates and assigned users of a task.

Member: - Represents an individual user of the system. Contains the users email address and name.

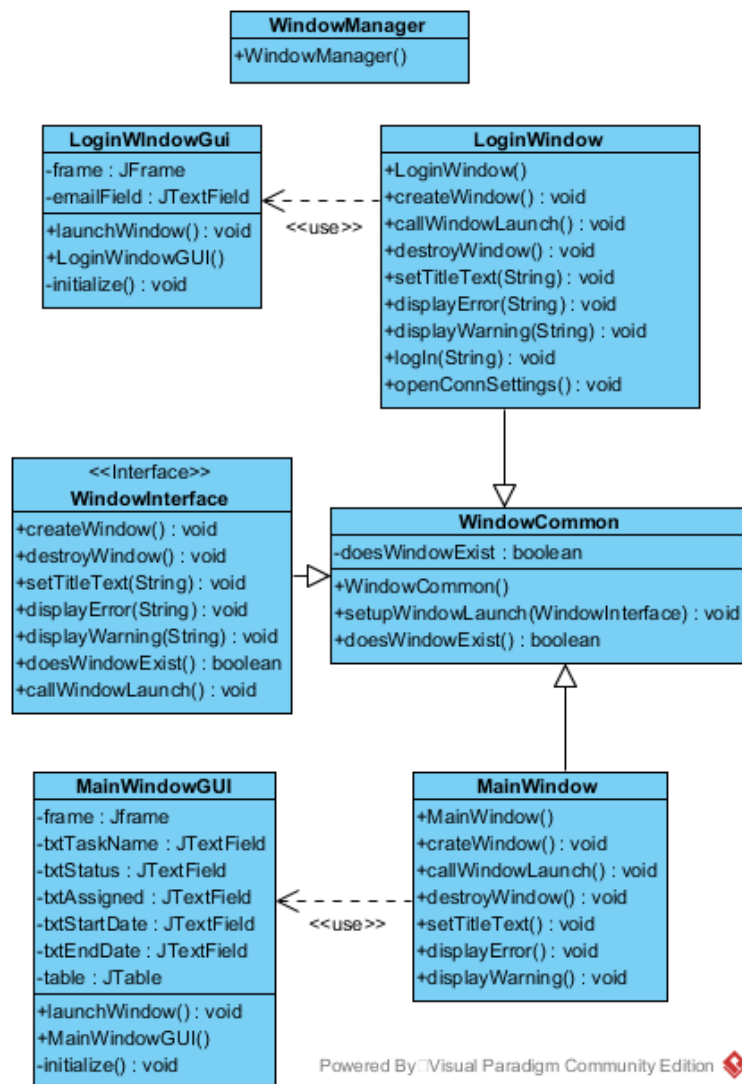
DelayTimer: - Provides a wrapper around the Timer class provided by Java. Used to queue and trigger actions such as database synchronisation.

MemberList: - A class which holds all members found in TaskerSRV as Member objects.

TaskList: - A class which holds all tasks found in TaskerSRV as Task objects.

TaskElement: - Represents a single element of a Task.

5.1.2 GUI Class Diagram



5.1.3 Logic Classes Interface List

Database:

```

void saveUsernames(String filePath);
bool connect(String URL, String filePath);
bool closeDbConn();
connStatus setConnStatus();
Task[] getTasks(String username);
Long getLastSyncTime();
Long System.Time.getmillis()
Members[] getMembers();
  
```

Task:

```

void SetTaskStatus(TaskStatuses Status);
void SetTaskElements();
void SetTask(String);
  
```

```
bool TaskChanged();  
bool Equal (object 0);  
void SetTaskChanged(bool hasChanged);
```

Member:

```
String getName();  
String getEmail();  
void setName();  
void setEmail();
```

DelayTimer:

```
Void ScheduleTask(TimerTask toRun, Long delay);
```

MemberList:

```
Member[] all members;  
void saveMembers (String filePath);  
members[] loadMembers(String filePath);  
verifyMember(String username);
```

TaskList:

```
Member getOwner();  
Task[] getAssignedTasks();  
void setOwner();  
void setAssignedTasks();
```

TaskElement:

```
void setElementName();  
void setElementComment();  
String getElementName();  
String getElementComment();
```

5.1.4 GUI Classes Interface List

Window:

```
void populateWindowArray();  
void initialize (windowIndex, windowName);  
void exit(windowIndex, windowName);  
void setFocus(windowIndex, windowName);
```

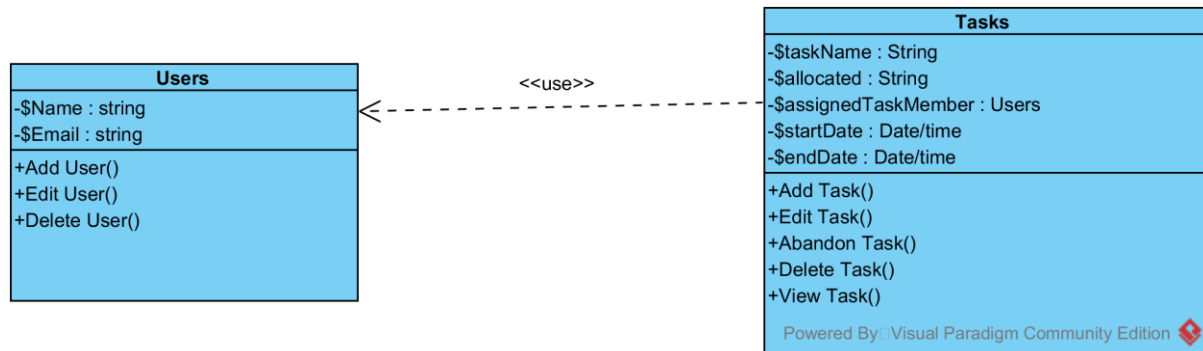
5.2 TaskerMAN

Programming using object oriented paradigms is new in PHP. A balance was struck up between procedural functions and object with methods to overcome the limitations of using object orientation within PHP.

5.2.1 Object Oriented Classes

The Users class holds relevant team member data which is received from TaskerSRV and is used to track users assigned to tasks as described below.

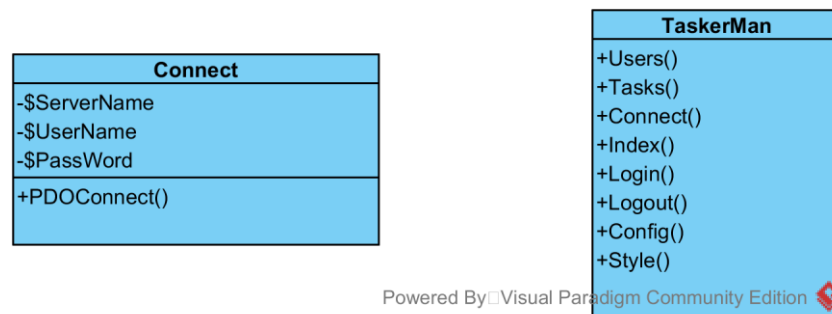
Tasks are also represented as by the Task class; these hold a reference to the assigned user to allow tracking of task allocation as required by FR4.



5.2.2 Procedural Classes

A Connect class holds the database connection internally and allows queries to be sent to a TaskerSRV database. The implementation is provided by PHP PDO.

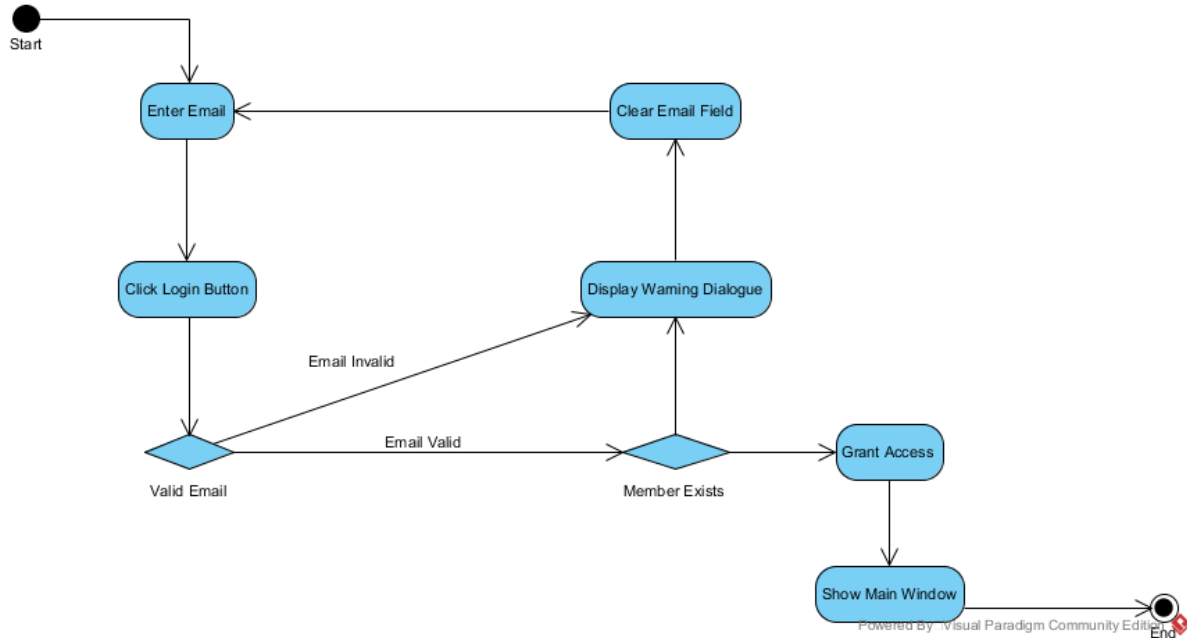
TaskerMAN is a PHP script which holds various procedural functions required by TaskerMAN



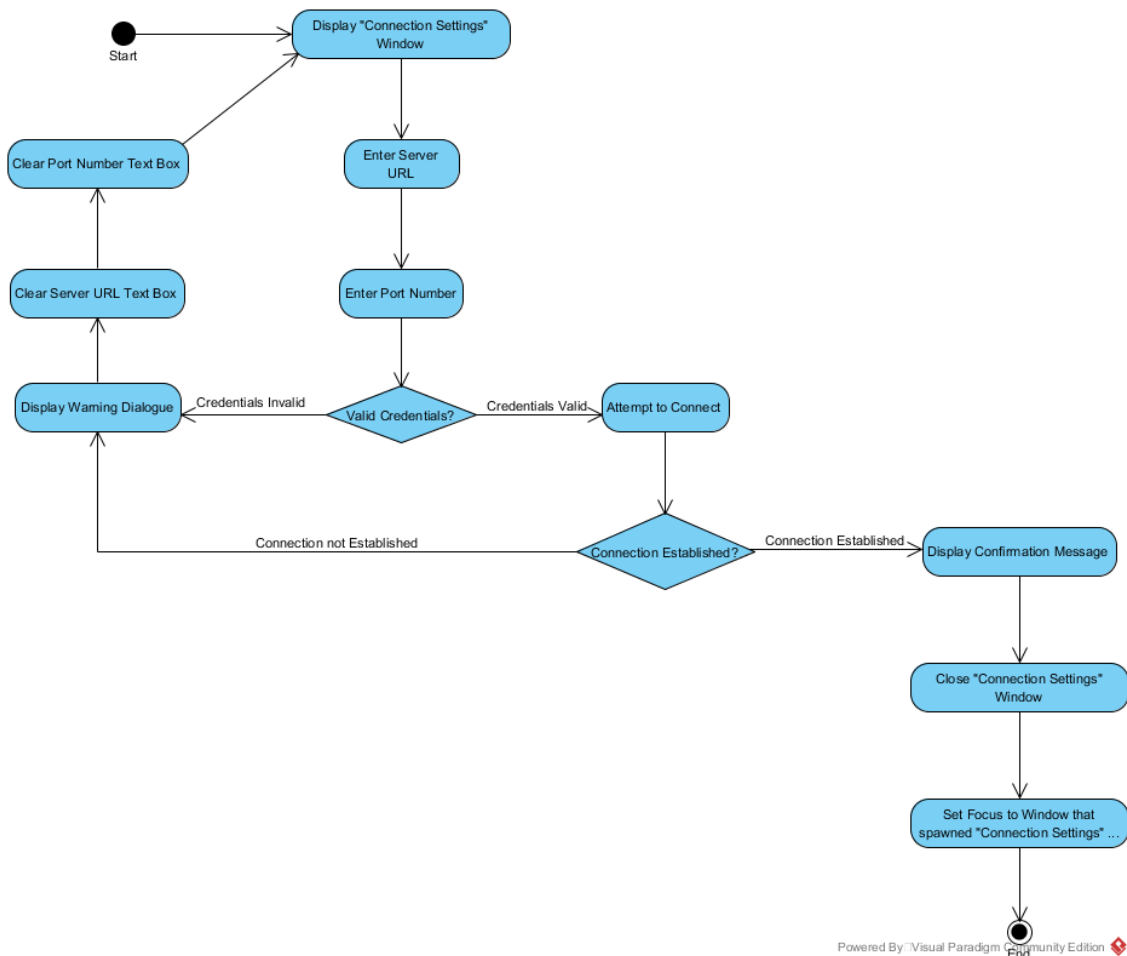
6. DETAILED DESIGN

6.1 Activity Diagrams

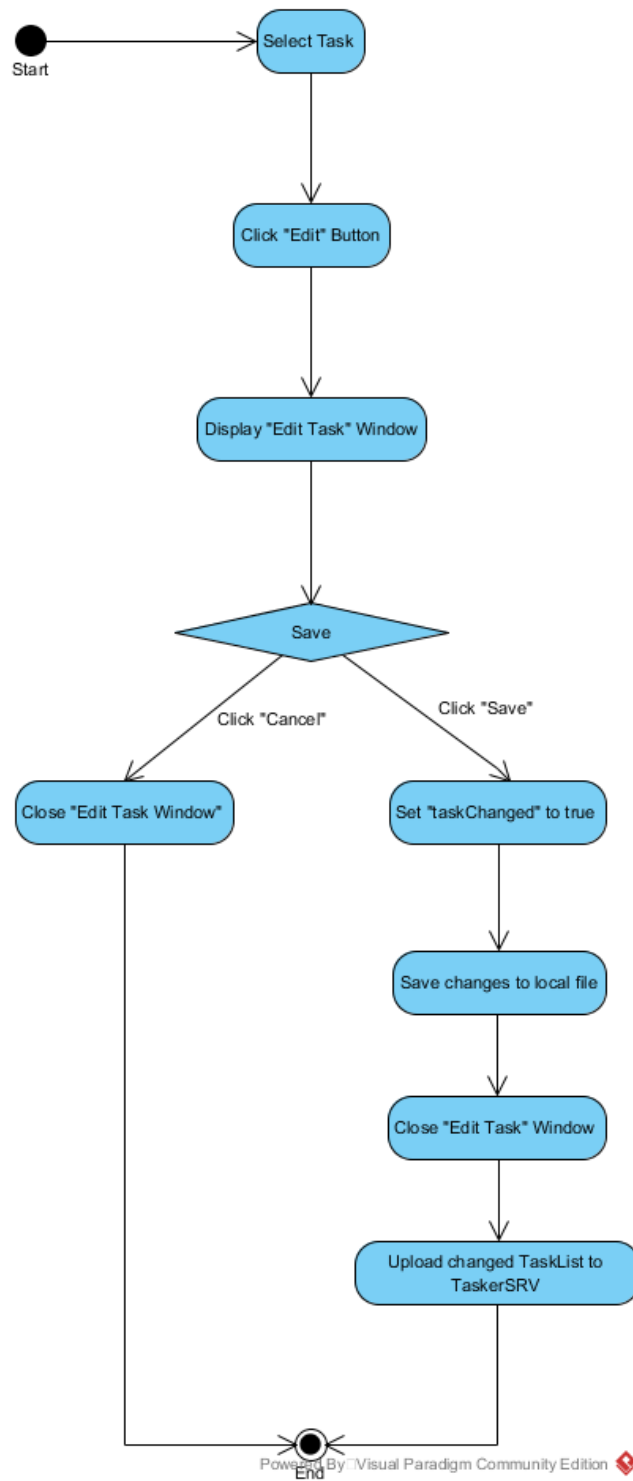
6.1.1 TaskerCLI User Login



6.1.2 TaskerCLI connecting to TaskerSRV

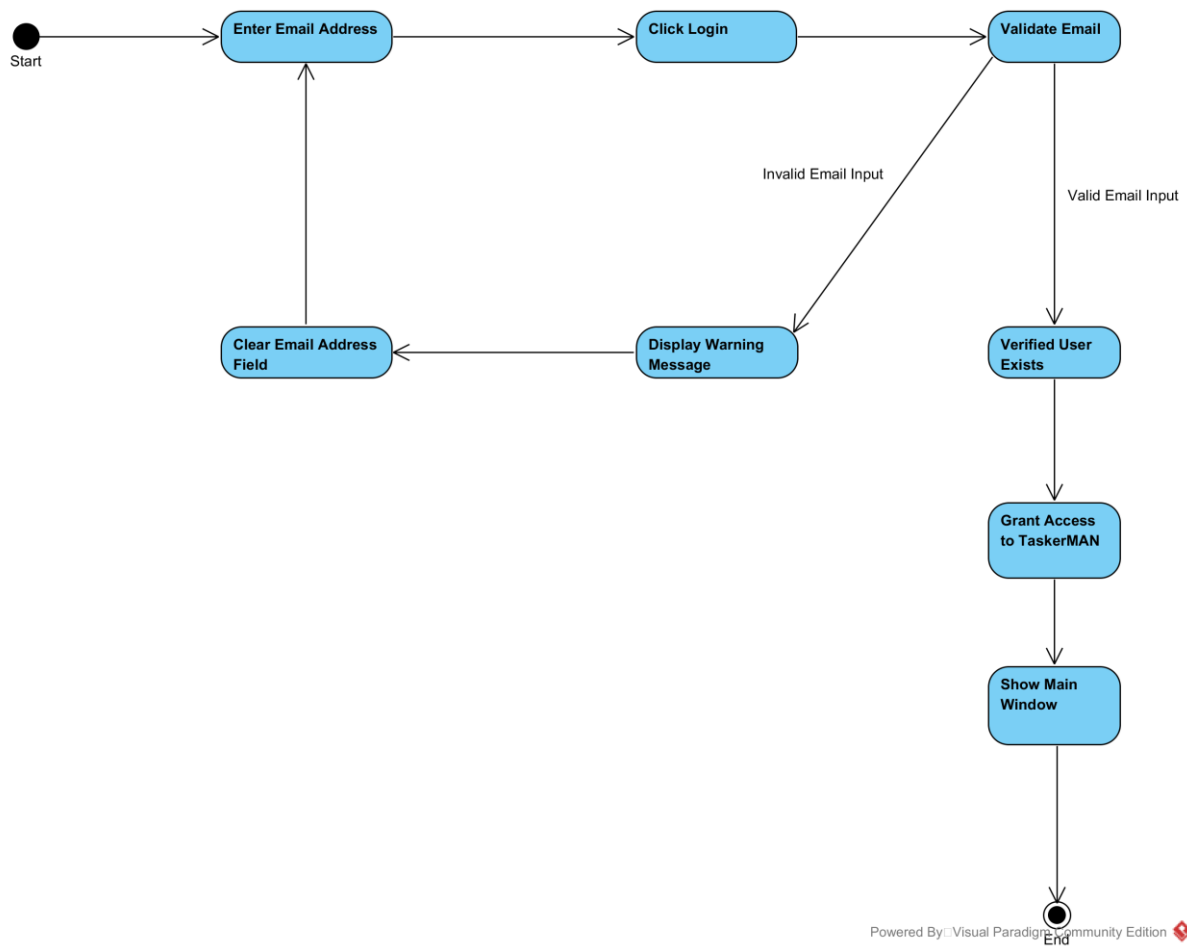


6.1.3 TaskerCLI Task Editing



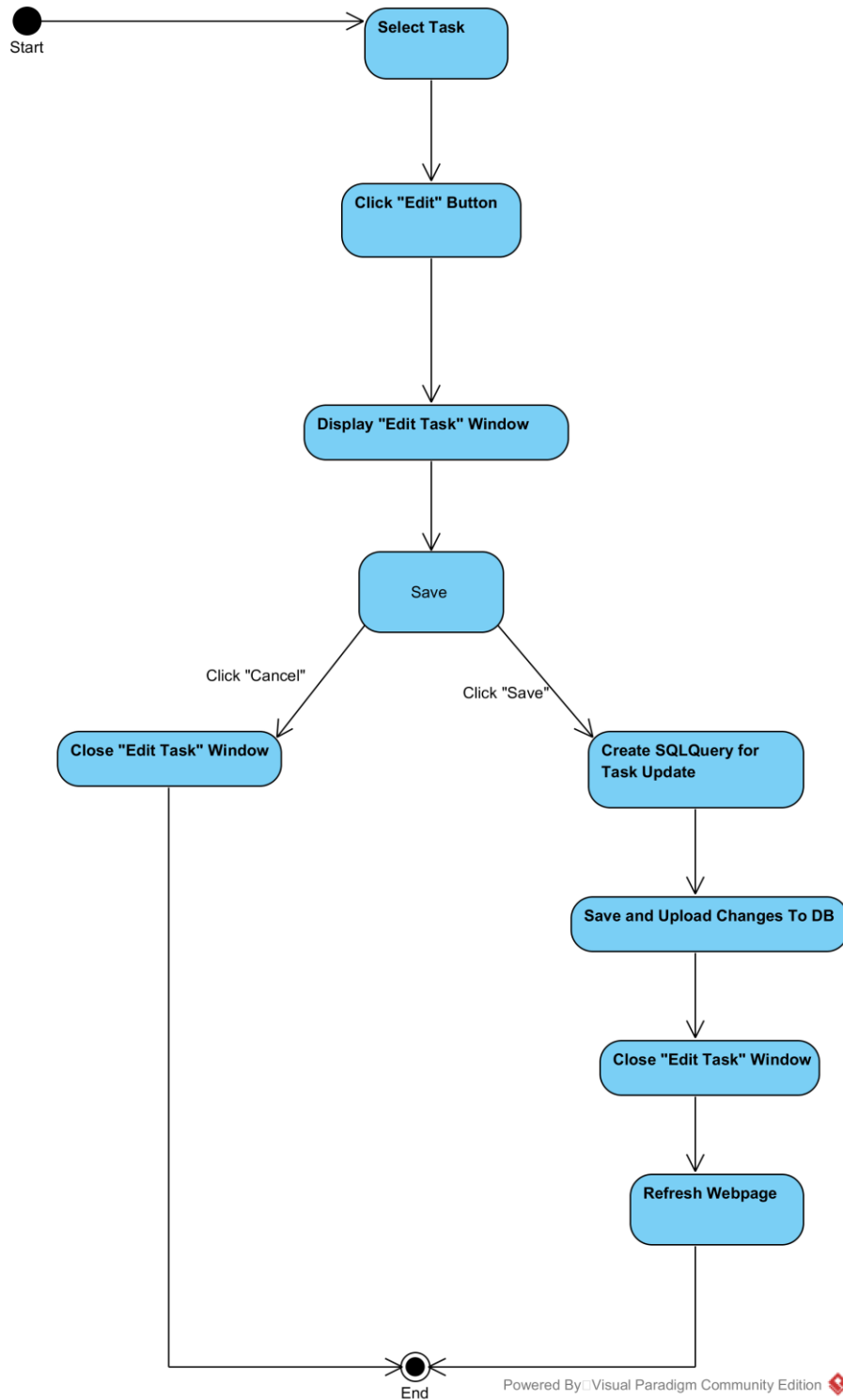
6.1.4 TaskerMAN Login

The login process for TaskerMAN was decomposed into discrete steps which are shown in the diagram below. This then displays the main window specified in FR7 to the user.



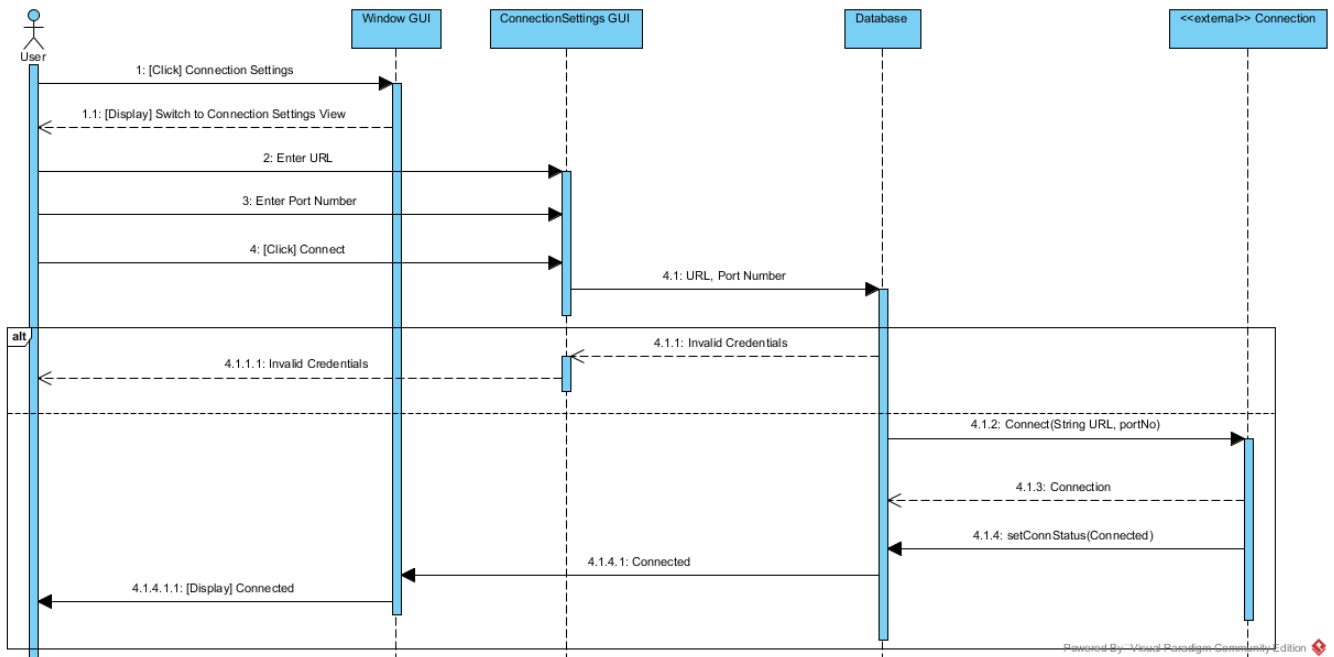
6.1.5 Editing Tasks in TaskerMAN

TaskerMAN must be able to edit tasks to allow reallocation or maintenance of task data as specified by requirement FR5. This diagram below lists the activities which take place when a task is being edited.

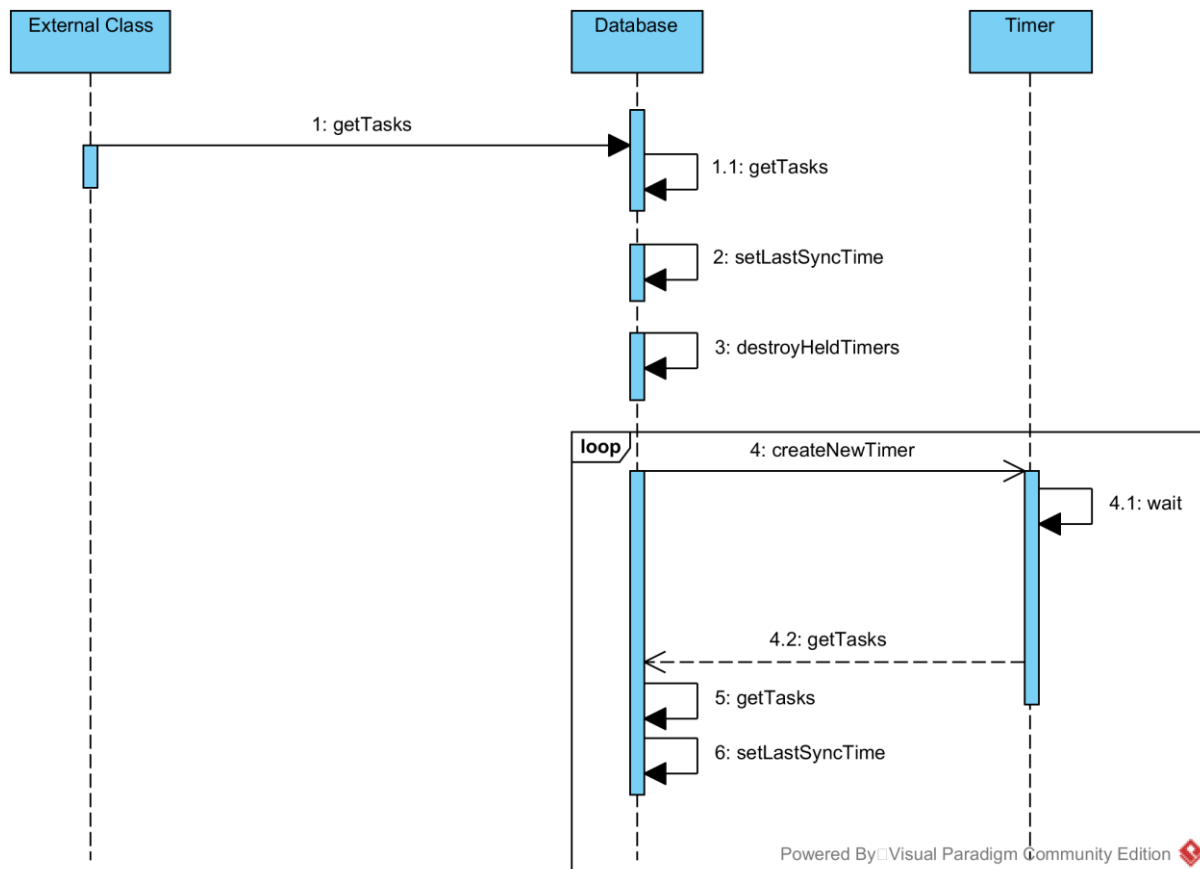


6.2 Sequence Diagrams

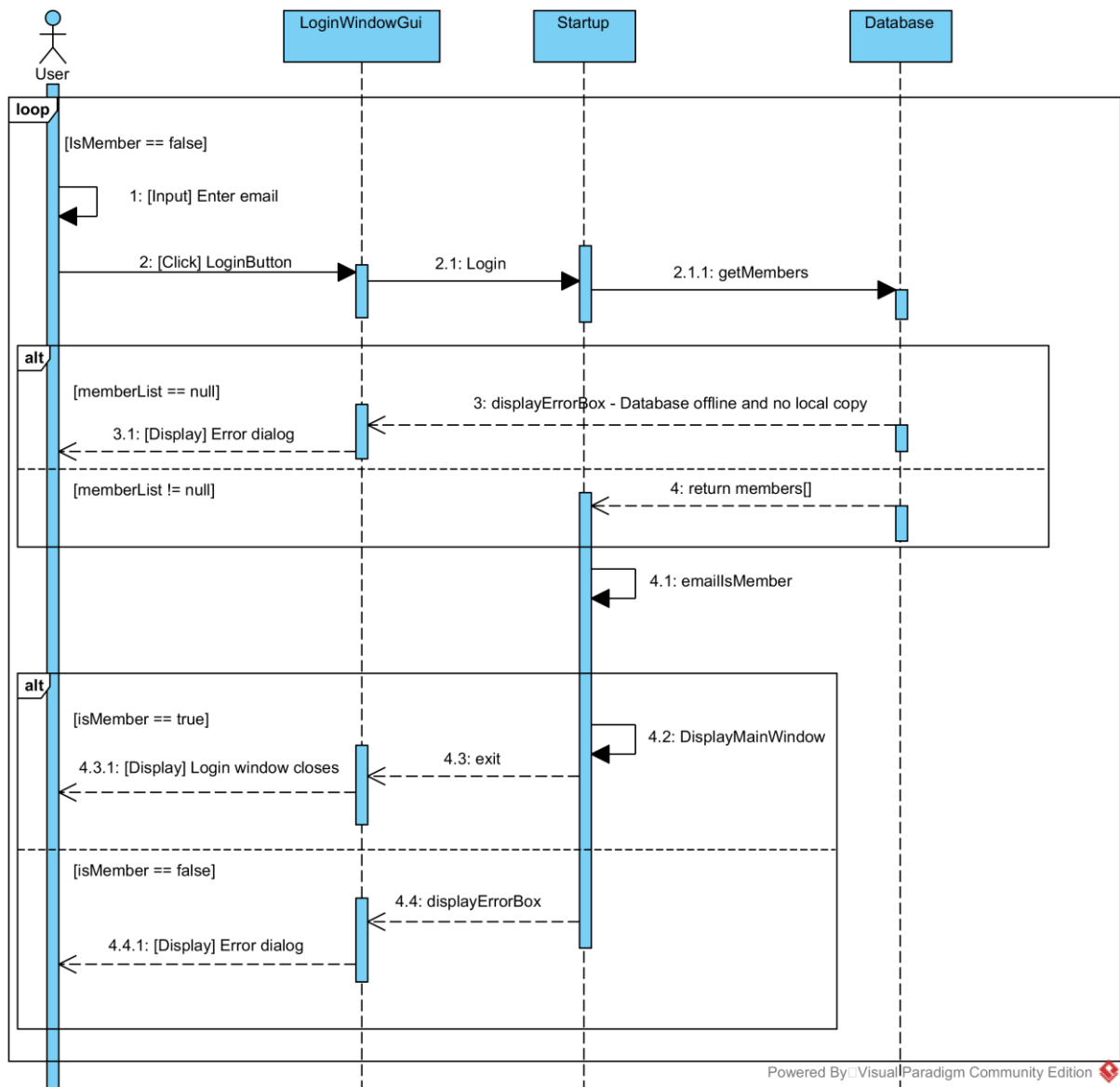
6.2.1 Connecting to TaskerSRV from TaskerCLI



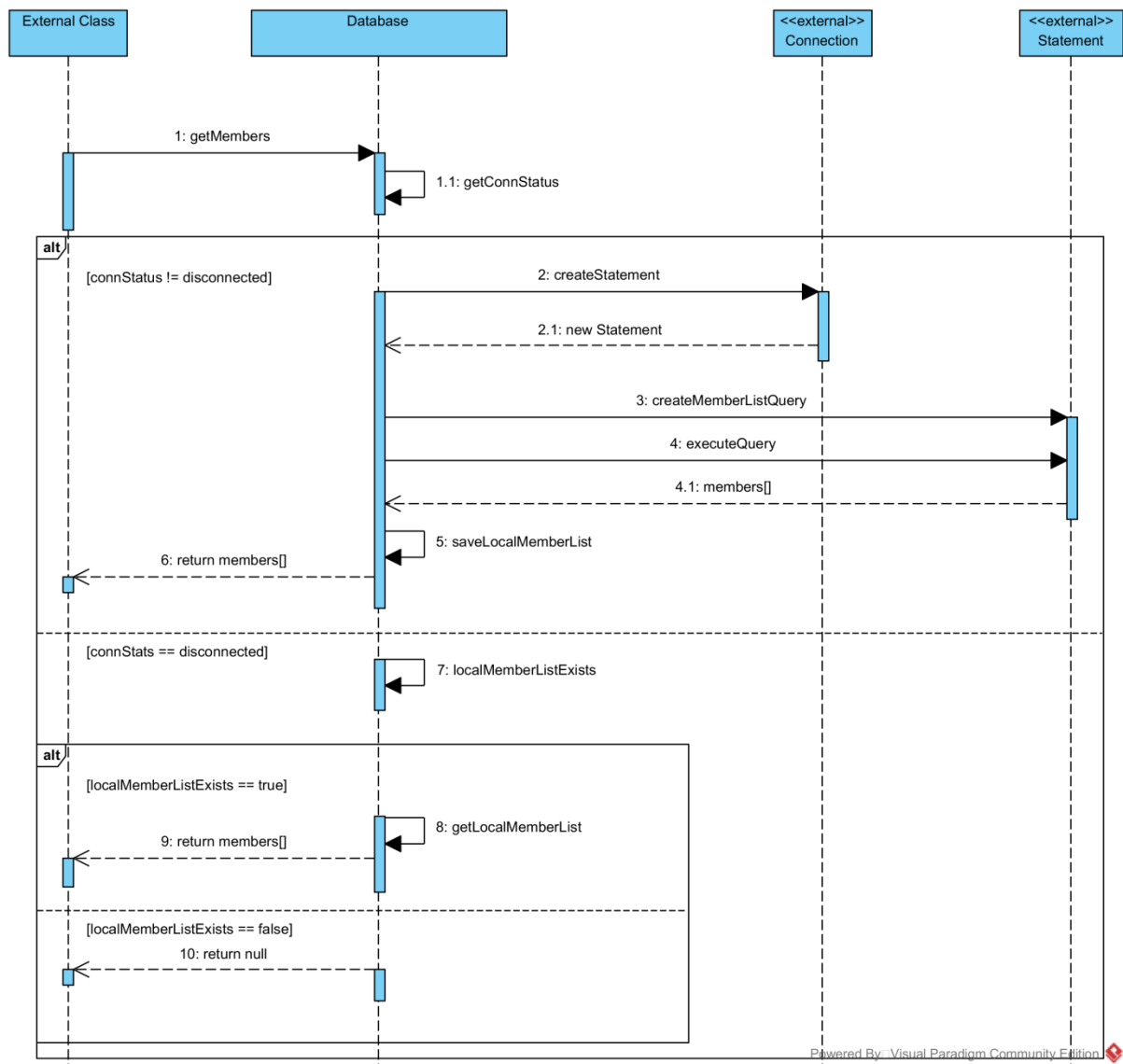
6.2.2 Automatic synchronization to TaskerSRV from TaskerCLI



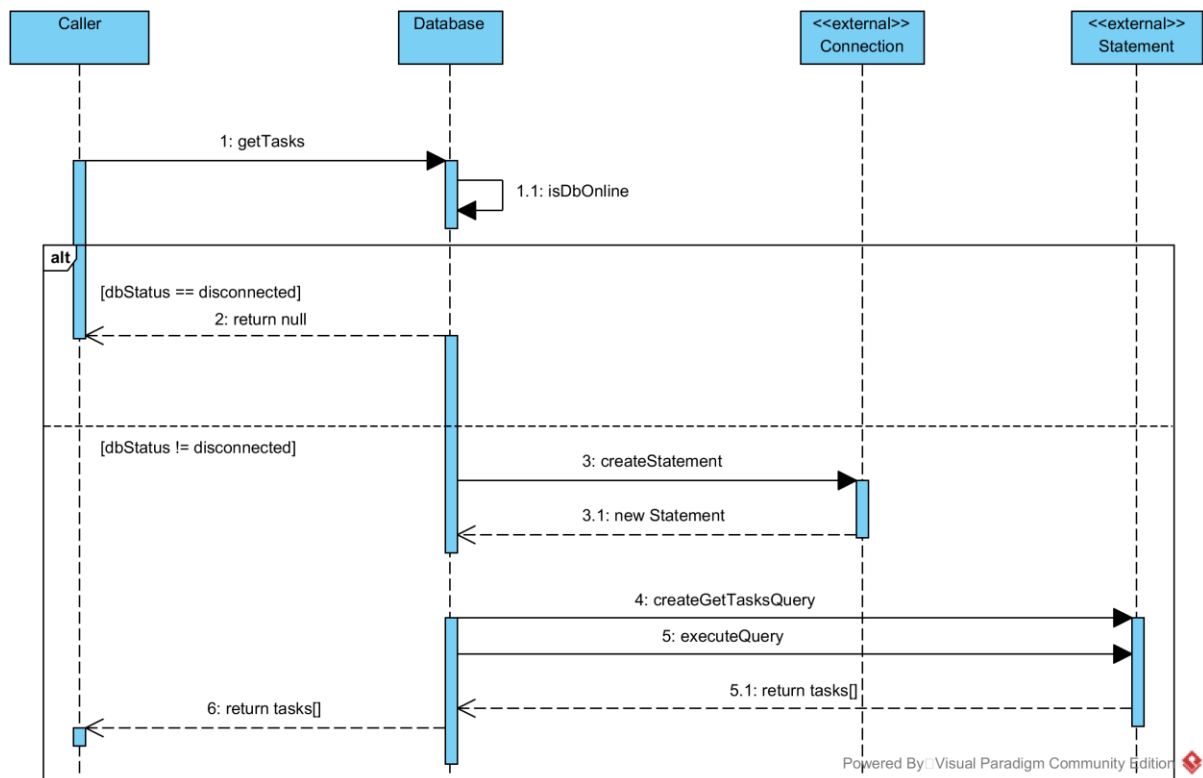
6.2.3 User Login to TaskerCLI



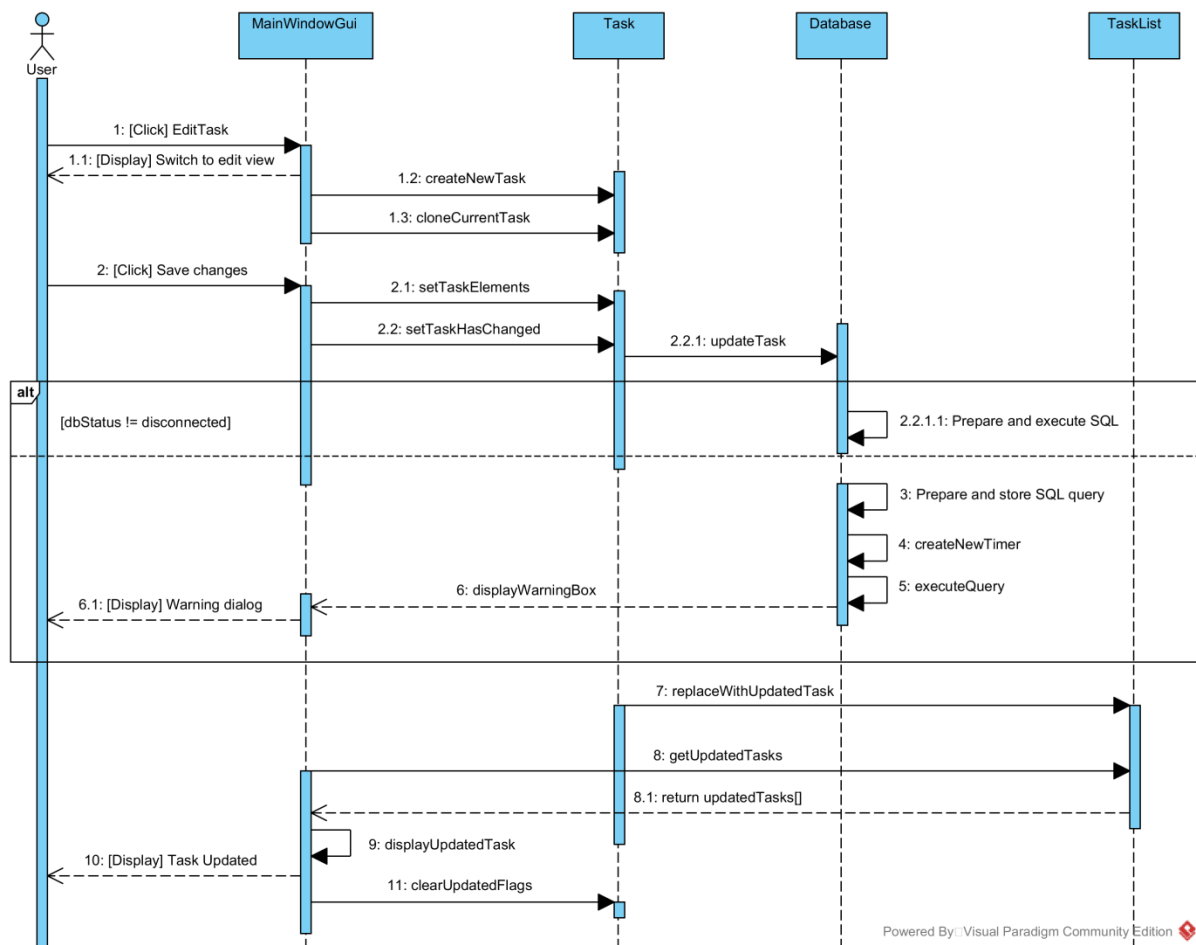
6.2.4 Get list of members in TaskerCLI from TaskerSRV



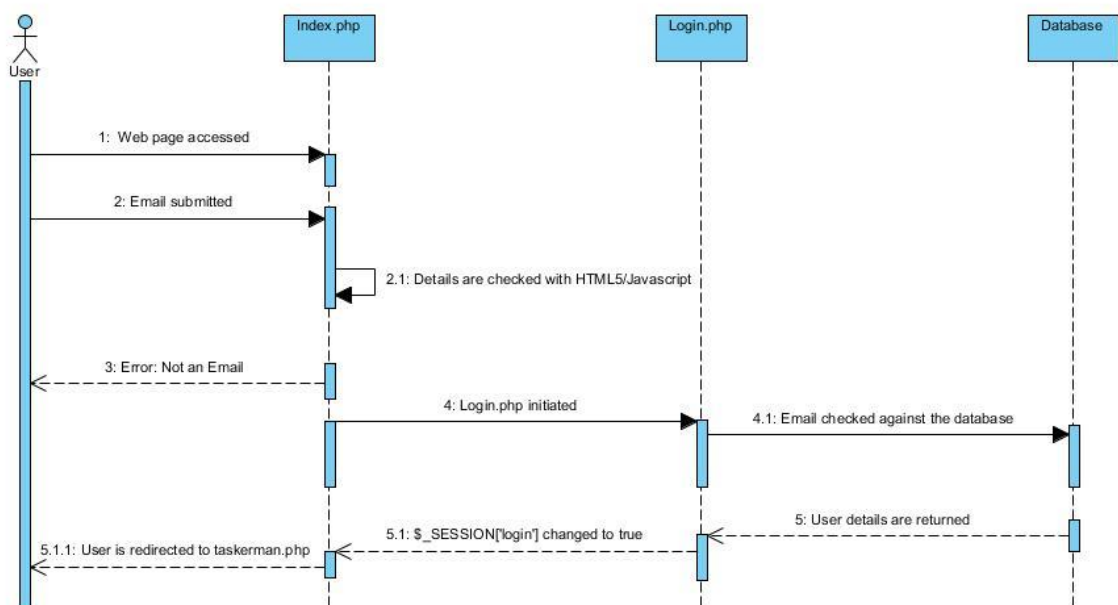
6.2.5 Get list of tasks in TaskerCLI from TaskerSRV



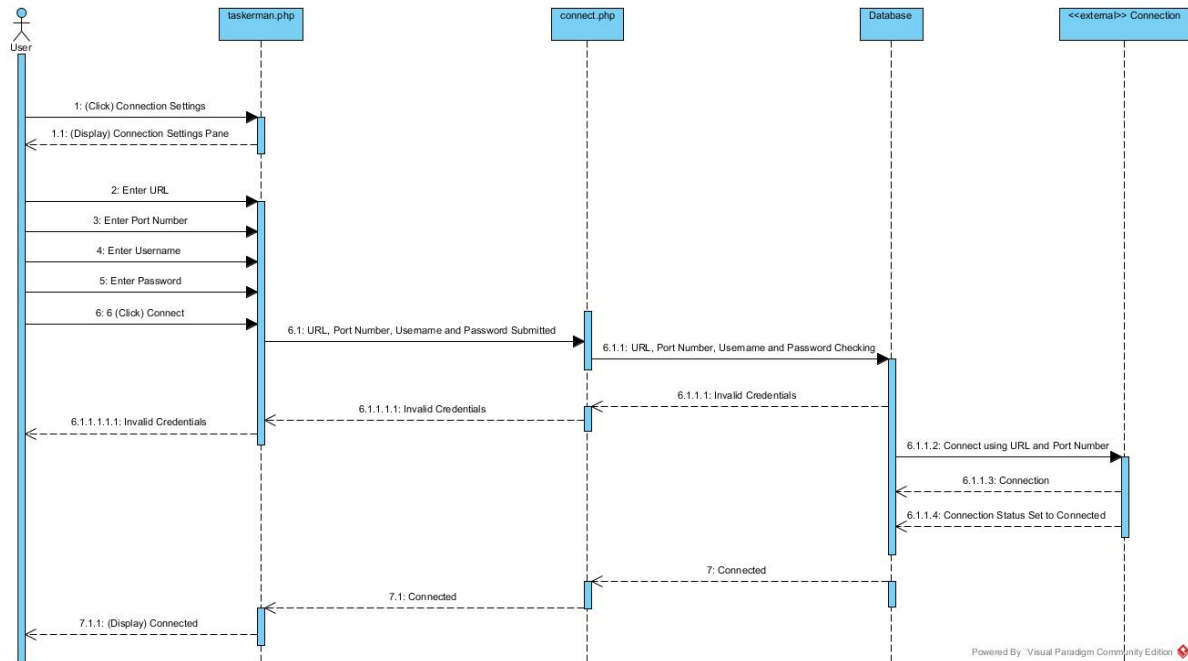
6.2.6 Editing tasks in TaskerCLI



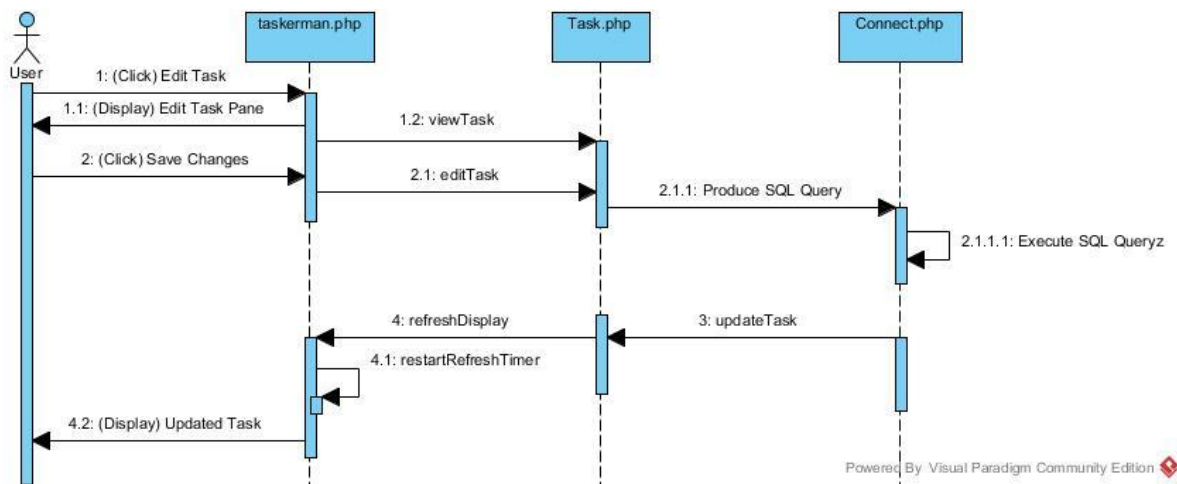
6.2.7 User Login in TaskerMAN



6.2.8 Connecting to TaskerSRV from TaskerMAN



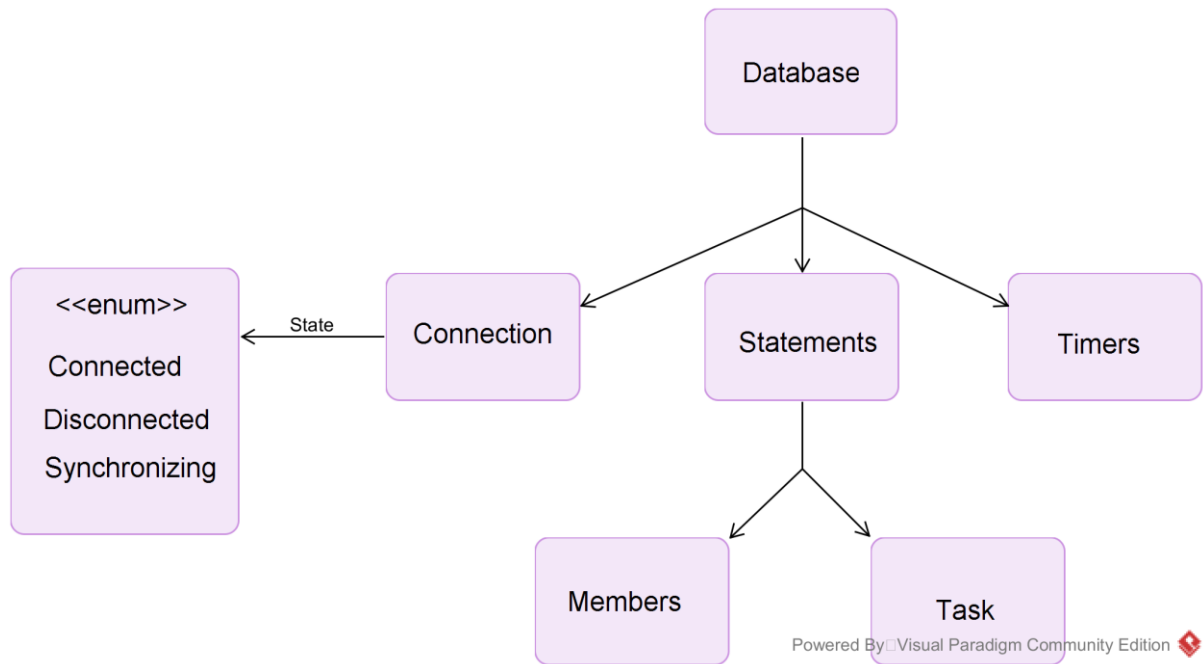
6.2.9 Editing Tasks in TaskerMAN



6.3 TaskerCLI Data Structures

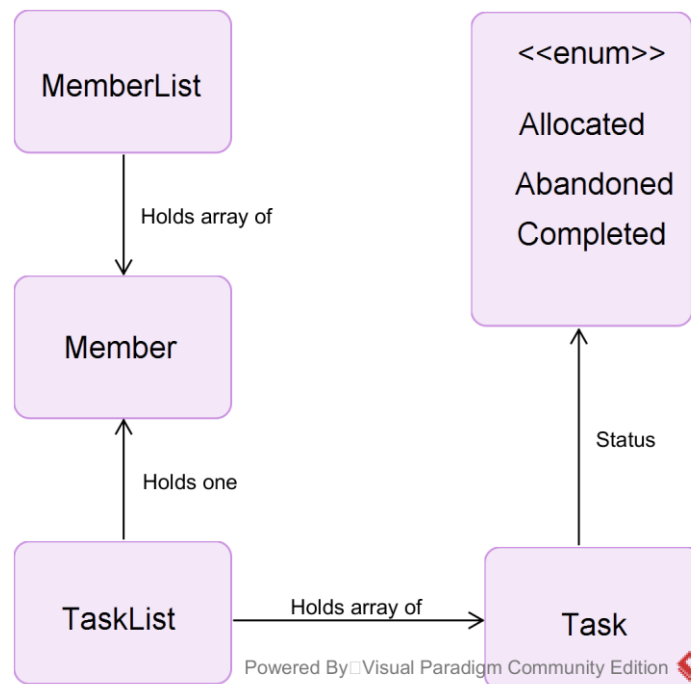
6.3.1 Database Structure

The diagram below shows the data classes the Database class uses in order to function



6.3.2 Members and Tasks Structure

The below diagram shows the hierarchy of Member related classes and Task related classes. From this diagram it is obvious the number and type of objects a class will hold or use.



6.4 Spike Work

6.4.1 TaskerCLI

The largest risk to TaskerCLI was implementing GUI's. This was identified as a significant risk early in planning so spike work was created to assess and mitigate any issues. For this spike work several people created the mock ups in Java using various tools.

Window builder allowed us to rapidly and accurately create the required design for the GUI; however the designer can be unintuitive with minor changes requiring major layout method changes. The group met and discussed finding and identified common pitfalls and how to avoid them, such as using several frames where layout could change.

In the process we established that by using an interface on all window classes we could easily construct and destruct these windows whilst abstracting over the minor differences between them.

This concept leads to creating a window manager which handles opening and closing windows on behalf of the software. Refining the code the final design was an array of window interface objects. By using an enumerated list we can access windows and manipulate them by doing a single line such as `"windowManager.createWindow("MAIN_WINDOW_ENUM");"`

6.4.2 TaskerMAN

Once PHP was chosen to drive TaskerMAN the team looked into PHP Unit. Having all used and understood the Java equivalent Junit the group needed to check if this knowledge was applicable to PHP Unit for creating unit tests.

This spike work consisted of setting up and learning how to implement PHP unit tests. One outcome which came to light was that our planned usage for PHP is mostly procedural whilst PHP unit caters for object oriented paradigms. The group also had to train each other in configuring the IDE to correctly implement these tests and writing suitable tests in PHP Unit.

Additional spike work on input validation was performed for TaskerMAN. As emails are used as an input we needed to check the input before sending it to the server. Several approaches were discussed such as using HTML 5 and JavaScript tests to detect invalid input on the client side. The approach decided upon was regular expression tests to ensure input is valid.

REFERENCES

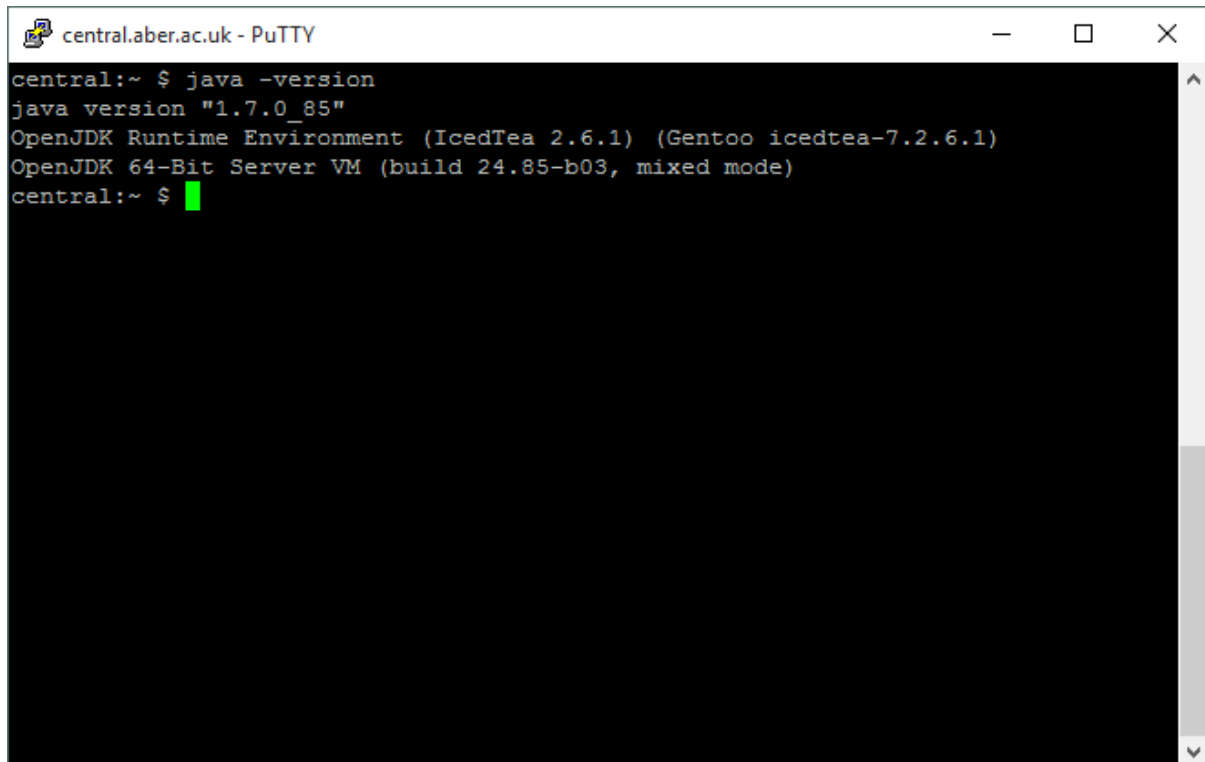
- [1] N. W. Hardy, *Tasker Team Tasking System - Requirement Specification 1.2*, Aberystwyth University: Software Engineering Group Project, 2015.
- [2] Tutorialspoint, "TutorialsPoint JUnit Environment Setup," Tutorialspoint, 27 10 2015. [Online]. Available: http://www.tutorialspoint.com/junit/junit_environment_setup.htm. [Accessed 27 10 2015].
- [3] Earl, Oliver; The PHP Group, "PHPInfo running on Apache," 27 10 2015. [Online]. Available: <http://users.aber.ac.uk/ole4/phpinfo.php>. [Accessed 28 10 2015].
- [4] S. Bergmann, "PHPUnit English Documentation," 28 10 2015. [Online]. Available: <https://phpunit.de/manual/current/en/phpunit-book.html#installation.requirements>. [Accessed 28 10 2015].
- [5] Oracle, "Oracle Software Delivery Cloud - MySQL Standard Edition for Linux x86-64," Oracle, 2015. [Online]. Available: https://edelivery.oracle.com/osdc/faces/SearchSoftware?_adf.ctrl-state=nmw6458k7_28&_afLoop=2752661125326430. [Accessed 21 10 2015].
- [6] N. W. Hardy, C. J. Price and B. P. Tiddeman, *SE.QA.05 A 1.8 - Design Specification Standards*, Aberystwyth University: Software Engineering Group Project, 2015.
- [7] Pixeden, "Firefox Web Browser Mockup Template," CorruptedDevelopment, 26 08 2014. [Online]. Available: <http://corrupteddevelopment.com/firefox-web-browser-mockup-template/>. [Accessed 26 10 2015].

DOCUMENT HISTORY

Version	CCF No.	Date	Changes made to document	Changed by
1.0	N/A	27/11/15	Original version	DAF5

APPENDICES

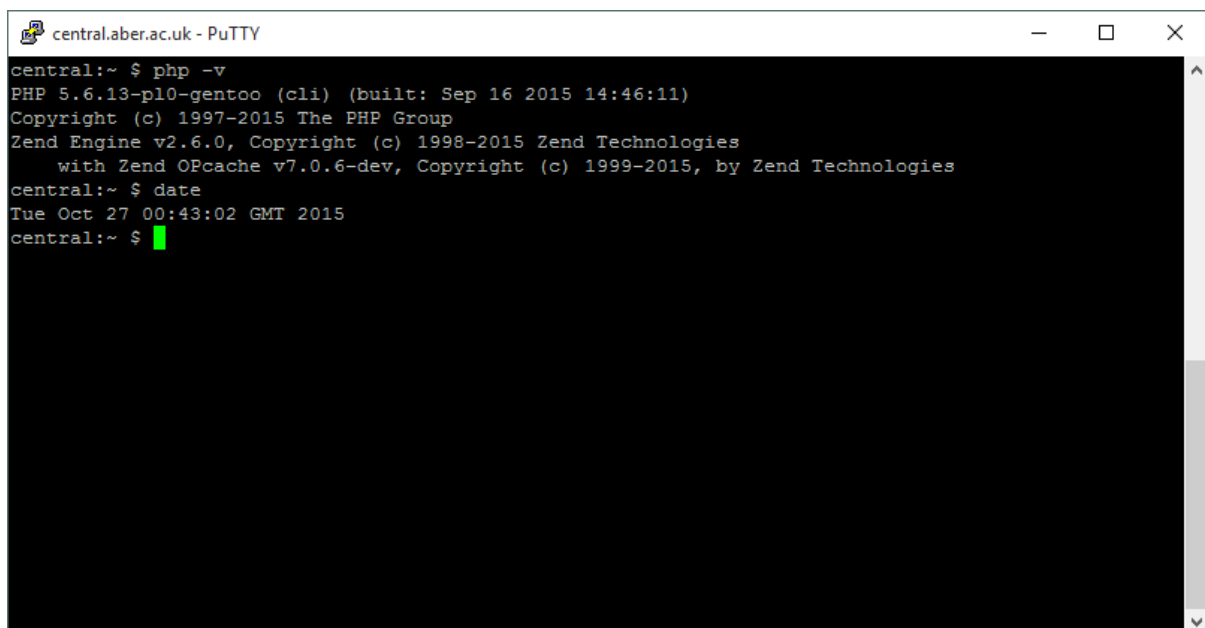
APPENDIX A – Java Version



A screenshot of a PuTTY terminal window titled 'central.aber.ac.uk - PuTTY'. The terminal shows the output of the command 'java -version'. The output is as follows:

```
central:~ $ java -version
java version "1.7.0_85"
OpenJDK Runtime Environment (IcedTea 2.6.1) (Gentoo icedtea-7.2.6.1)
OpenJDK 64-Bit Server VM (build 24.85-b03, mixed mode)
central:~ $
```

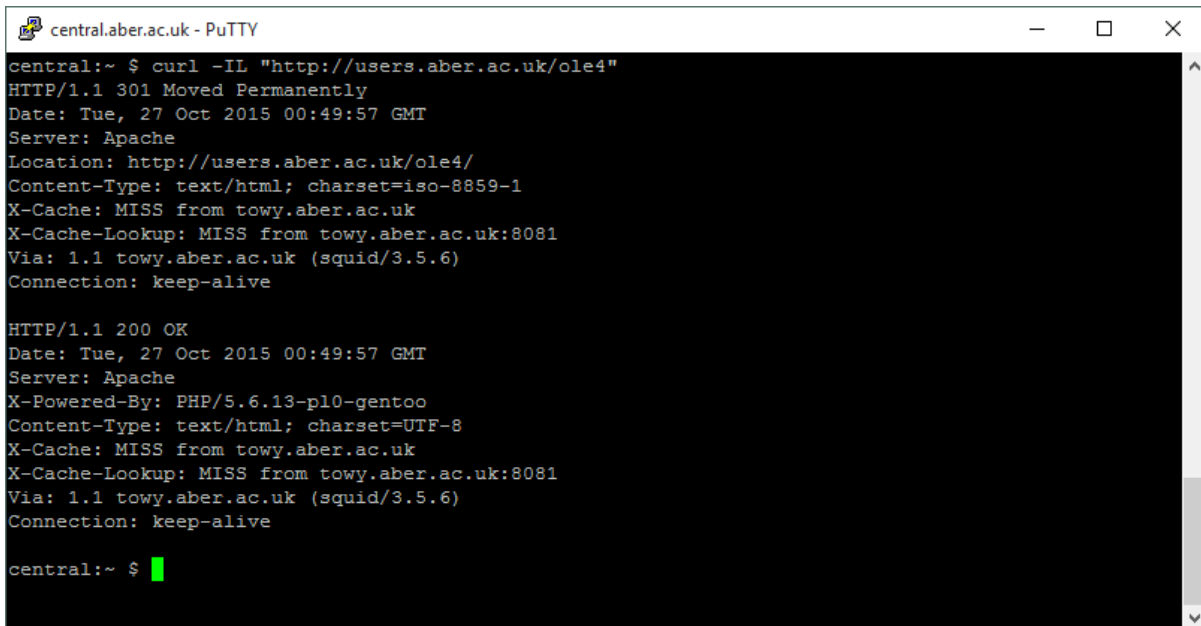
APPENDIX B – PHP Version



A screenshot of a PuTTY terminal window titled 'central.aber.ac.uk - PuTTY'. The terminal shows the output of the commands 'php -v' and 'date'. The output is as follows:

```
central:~ $ php -v
PHP 5.6.13-pl0-gentoo (cli) (built: Sep 16 2015 14:46:11)
Copyright (c) 1997-2015 The PHP Group
Zend Engine v2.6.0, Copyright (c) 1998-2015 Zend Technologies
    with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2015, by Zend Technologies
central:~ $ date
Tue Oct 27 00:43:02 GMT 2015
central:~ $
```

APPENDIX C – Apache Information

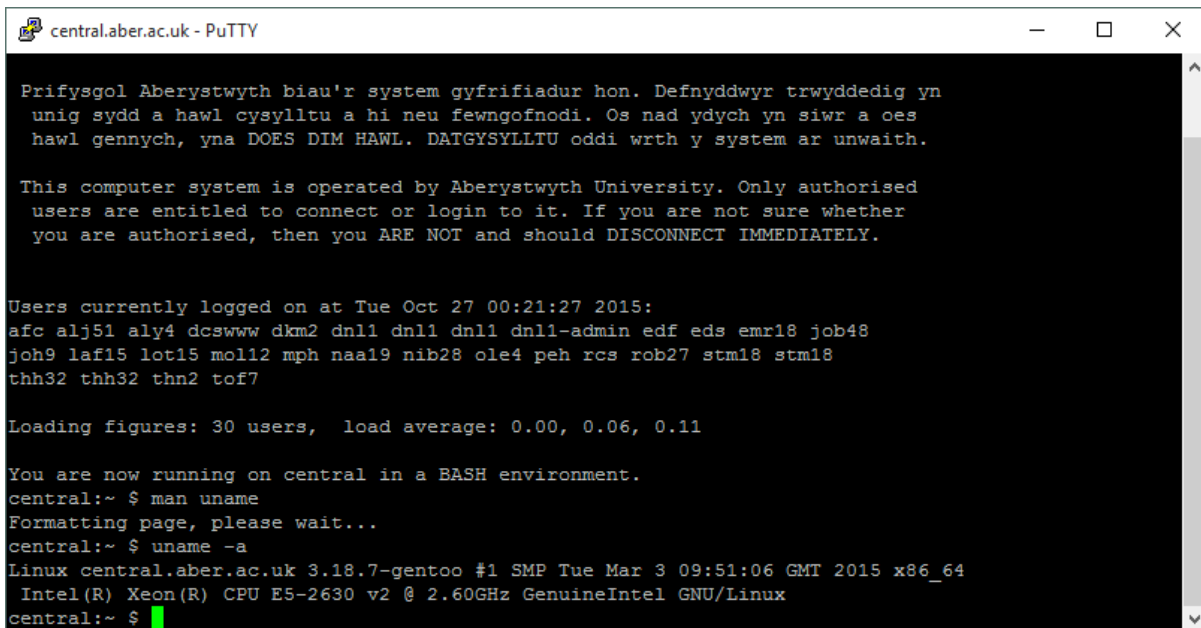


```
central.aber.ac.uk - PuTTY
central:~ $ curl -IL "http://users.aber.ac.uk/ole4"
HTTP/1.1 301 Moved Permanently
Date: Tue, 27 Oct 2015 00:49:57 GMT
Server: Apache
Location: http://users.aber.ac.uk/ole4/
Content-Type: text/html; charset=iso-8859-1
X-Cache: MISS from towy.aber.ac.uk
X-Cache-Lookup: MISS from towy.aber.ac.uk:8081
Via: 1.1 towy.aber.ac.uk (squid/3.5.6)
Connection: keep-alive

HTTP/1.1 200 OK
Date: Tue, 27 Oct 2015 00:49:57 GMT
Server: Apache
X-Powered-By: PHP/5.6.13-pl0-gentoo
Content-Type: text/html; charset=UTF-8
X-Cache: MISS from towy.aber.ac.uk
X-Cache-Lookup: MISS from towy.aber.ac.uk:8081
Via: 1.1 towy.aber.ac.uk (squid/3.5.6)
Connection: keep-alive

central:~ $
```

APPENDIX D – Linux information



```
central.aber.ac.uk - PuTTY

Prifysgol Aberystwyth biau'r system gyfrifiadur hon. Defnyddwyr trwyddedig yn
unig sydd a hawl cysylltu a hi neu fewngofnodi. Os nad ydych yn siwr a oes
hawl gennych, yna DOES DIM HAWL. DATGYSYLLTU oddi wrth y system ar unwaith.

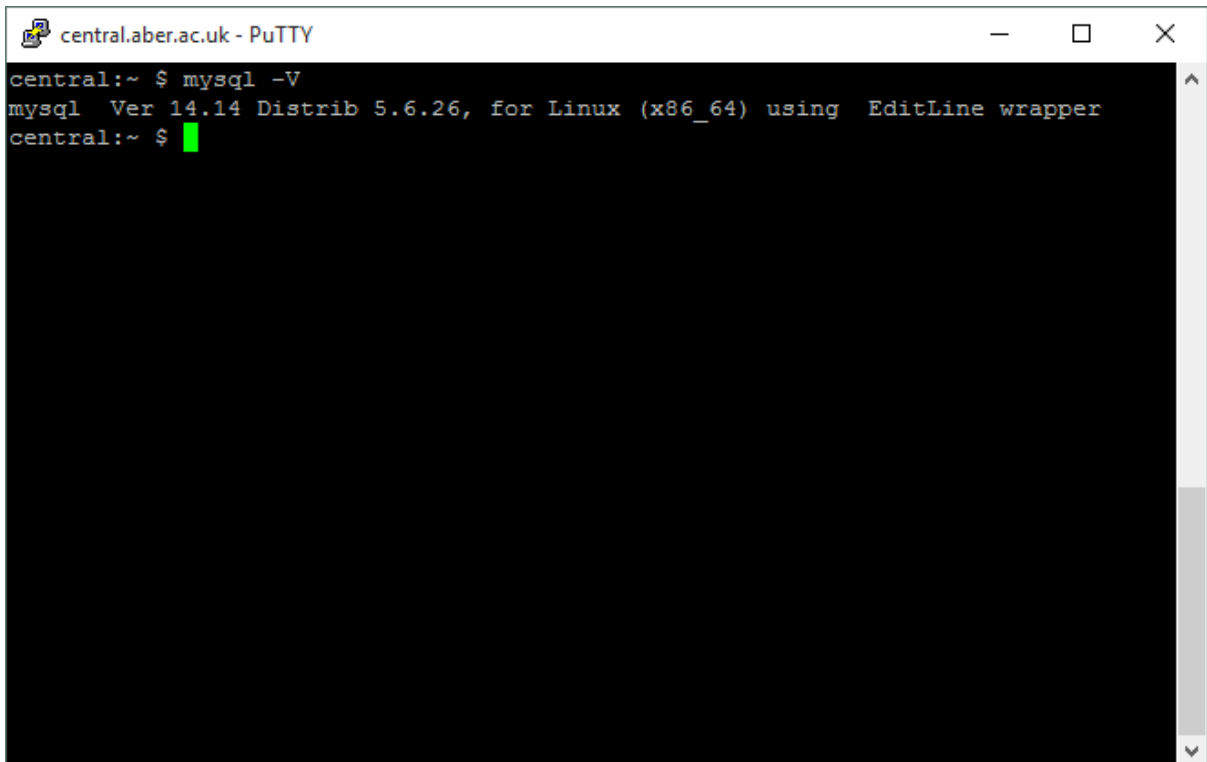
This computer system is operated by Aberystwyth University. Only authorised
users are entitled to connect or login to it. If you are not sure whether
you are authorised, then you ARE NOT and should DISCONNECT IMMEDIATELY.

Users currently logged on at Tue Oct 27 00:21:27 2015:
afc alj51 aly4 dcswww dkm2 dn11 dn11 dn11 dn11-admin edf eds emr18 job48
joh9 laf15 lot15 mol12 mph naa19 nib28 ole4 peh rcs rob27 stm18 stm18
thh32 thh32 thn2 tof7

Loading figures: 30 users,  load average: 0.00, 0.06, 0.11

You are now running on central in a BASH environment.
central:~ $ man uname
Formatting page, please wait...
central:~ $ uname -a
Linux central.aber.ac.uk 3.18.7-gentoo #1 SMP Tue Mar 3 09:51:06 GMT 2015 x86_64
Intel(R) Xeon(R) CPU E5-2630 v2 @ 2.60GHz GenuineIntel GNU/Linux
central:~ $
```

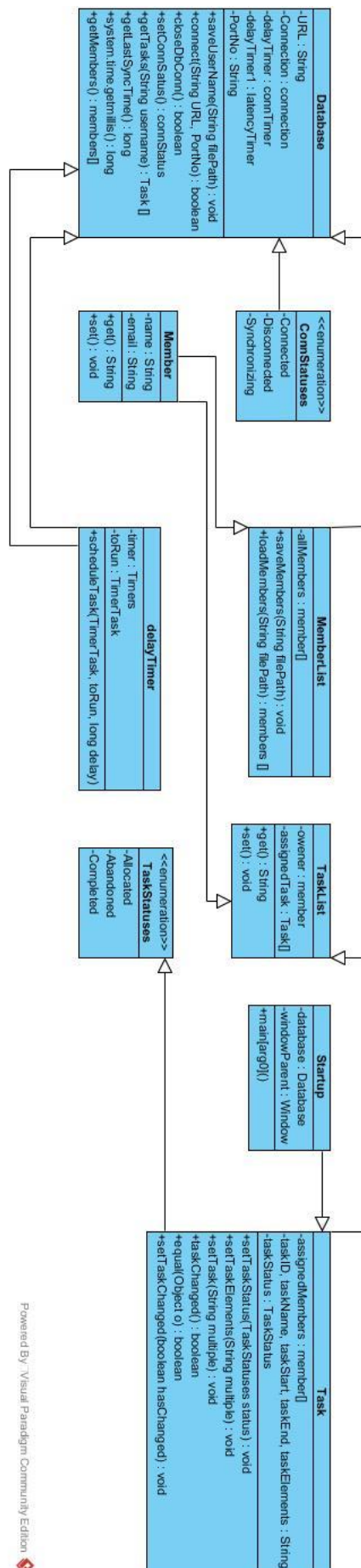
APPENDIX E – MySQL Version



The image shows a PuTTY terminal window titled "central.aber.ac.uk - PuTTY". The terminal output shows the command `mysql -V` being executed, resulting in the output: `mysql Ver 14.14 Distrib 5.6.26, for Linux (x86_64) using EditLine wrapper`. The prompt `central:~ $` is visible on the line below the output.

```
central.aber.ac.uk - PuTTY
central:~ $ mysql -V
mysql Ver 14.14 Distrib 5.6.26, for Linux (x86_64) using EditLine wrapper
central:~ $
```

APPENDIX F – Logic Class Diagram



Powered By Visual Paradigm Community Edition

