**Penalty Points Bureau**

Colin Maher

<https://github.com/SomeIrishLad/PPB>

Logins

Username: admin Username: ppbemp2

Password: password Password: password2

Username: ppbemp1 Username: ppbemp3

Password: password1 Password: password3

Admin Email Server Email (SMTP)

[penaltypointsbureau@gmail.com](mailto:penaltypointsbureau@gmail.com) [ppbserver@gmail.com](mailto:ppbserver@gmail.com)

ppbpass123 ppbpass123

Overview

This software was designed for the fictional government agency “Penalty Points Bureau”. It does not contain much functionality on the main screen as this was not in the brief. However all the necessary background functionality to quickly and easily expand the application has been implemented.

Github’s integration with Visual studio was used from the start. It helped to keep track of changes and roll back to older versions of the project if needed.

The client was designed using Microsoft’s Model-View-ViewModel design pattern. This allows complete separation of the UI and background logic. The UI is linked to the viewmodel using data binding and commands. This makes the project easy to maintain, test and it can be easily ported to a web application or perhaps even mobile.

All communication between the client and server uses json and is encrypted.

The server deals with all database interactions and returns results to the client.

All commands from the client to the server and vice versa use the ServerCommand class which can be found in Models. A server command contains a string e.g “login\_user” which tells the other end what the command is for. It also contains a dictionary which stores key value pairs of whatever data is being sent. This is very extensible, with little effort I can create a new command for a completely different use. E.g Add user to db etc.

Use Cases

1. User Login

**Actor:** PPB Employee

**Entry conditions:**

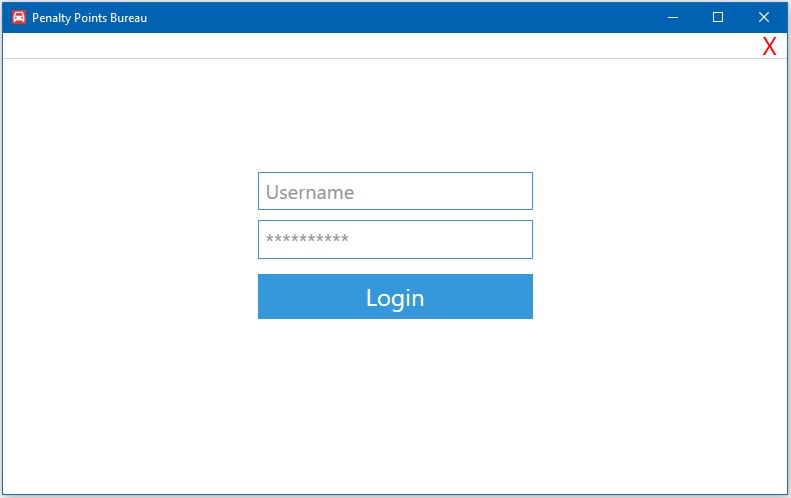
A user wishes to log in to the system. They run the client program.

**Flow of events:**

1. The system presents a login screen which requests:
   1. Username
   2. Password
2. The user enters their username and password clicks the login button.
3. The client creates a login command along with the entered username and password.
4. The command is serialised into a json string and encrypted.
5. The client sends the encrypted json to the server.
6. The server decrypts and deserialises the json.
7. The server checks that it is a login command and begins the process of logging in the user.
8. The server checks the number of failed login attempts by this user.
   1. 0 attempts. No action is taken.
   2. 1 -2 attempts. The server sends a message to the client with a warning of how many login attempts are remaining, which is then displayed to the user.
   3. 3 attempts. The server locks the user’s account preventing them from logging in.
9. The server checks the users table in the database for the entered username.
   1. The server returns the corresponding user id.
   2. The server fails to find the username. Notifies user of login failure.
10. The server appends the entered password to the user id. This string is hashed using SHA-256 for security reasons.
11. The user table is then checked for an entry with the correct user id and password-user id combo.
    1. This entry was found. The server prompts the client that login was successful. The client closes the login page and opens the main page.
    2. This entry was not found. The server prompts the client that login failed. The client notifies the user.

**Exit Conditions:**

The user has gains access to the system.



2. User Timeout

**Actor:** PPB Employee

**Entry conditions:** User is logged in

**Flow of events:**

1. User stops interacting the application.
2. The system starts the idle timer and sets it to 15 minutes.
   1. The user moves the mouse or presses a key before 15 minutes, which resets the timer to 15 minutes and starts the process again.
3. The user does not interact with the application for 15 minutes.
4. The application logs the user out and returns to the login screen.
5. The application notifies the server of the idle timeout.
6. The server logs it to file.

**Exit Conditions:**

The user is logged out.

3. Account Locked

**Actor:** PPB Employee

**Entry conditions:** User is attempting to login to system.

**Flow of events:**

1. User enters incorrect username/password
2. Each time the system notifies the user of how many login attempts are remaining.
3. On the third incorrect login, the user’s account is locked.
4. The server keeps track of how many login attempts each user has.
5. If they login successfully, their failed attempts get reset to 0.
6. The server will then refuse to login a user with 3 failed logins.
7. The user is notified to contact an admin. Who can reset it.

**Exit Conditions:**

User’s account is locked

4. Logout

**Actor:** PPB Employee

**Entry conditions:** User wishes to logout of the system.

**Flow of events:**

1. User clicks logout button.
2. The button sends a command which triggers the logout function.
3. The application is restarted which logs out the user.
4. The login page is displayed.
5. The server ends the session.

**Exit Conditions:**

The user is logged out and can safely leave the computer.

Copyright (c) 2017 Colin Maher

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.