Team Level Seven

Summer 2015

Pass Man: Secure Development Lifecycle

We have proposed to create a Java Application that serves as a password manager/tracker. The front-end UI and back-end logic will be written using Java 7, and the password data will be stored in a database (either MySQL or Oracle DBMS). The following is our analysis of Requirements and Design for our proposed application.

REQUIREMENTS

Security and Privacy Requirements:

Requirement	**Section**
System should prevent direct, unauthorized access (insert, alter, delete, query) to database.	Privacy/ Security
User should be able to create/manage account for sub-users (i.e. supervisor creating account for employee) to view certain information.	Privacy
Password data should be encrypted in backend.	Security
Audit trails for for changes to data should be maintained, including corresponding userid and timestamp.	Security
System should restrict access to an account after multiple failed access attempts.	Security

As security flaws arise, they will be documented and evaluated. The evaluation process will aim to determine the cost and/or difficulty of patching the flaw versus the security/privacy

risk that the flaw poses to the overall application (see bug bars, below). The risk factor of a flaw will have a higher priority in determining the urgency with which each flaw is to be addressed. For example, a security flaw that has a "moderate" difficulty factor and an "important" risk factor should be addressed with a higher priority than a "low" difficulty factor and a "moderate" risk factor. Even though the second flaw could be an easier/quicker fix, it does not pose as much of a threat to the system if left temporarily unaddressed while the first flaw is patched.

Bug Bars:

Security Bug Bar	
Critical	● Elevation of privilege ○ Remote anonymous user
Important	● Elevation of privilege ○ Remote authenticated user ○ Local authenticated user ● Information disclosure ○ System information ● Tampering ○ Permanent alteration of data that persists
Moderate	● Security assurances ○ Notification of security feature change(s)
Low	● Information disclosure ○ Runtime information ● Tampering ○ Temporary alteration of data that does not persist

Privacy Bug Bar

Critical	● Lack of data protection ○ Personal identifying information (PII) accessible without authentication
Important	● Lack of data protection ○ PII not protected by authentication mechanism
Moderate	● Lack of user controls ○ User unable to modify/delete stored PII
Low	● Lack of notice and consent ○ Storage of PII without notice and consent

Security and Privacy Risk Assessments:

Our program stores personally identifiable information (PII) on a user's computer, it is capable of transferring this data from the user's computer (P1), and has the ability to share personally identifiable information. Based on these facts we have assessed that our program needs to have a secure way of logging in to access the database of usernames and passwords but also a way to safely transfer this data too.

The type of personally identifiable information that we store and transfer are individual and shared usernames and passwords. Creating and remembering secure passwords is difficult especially if they are not pronounceable and use ambiguous characters. We want to make it easier for company employees to "remember" secure passwords without the hassle of trial and error or having to reset passwords. Some of the information stored are for a single employee's use only and others are shared passwords for multiple employees to use.

We believe that our program has value for businesses because usernames and passwords are hard to remember, especially if they are unique usernames and passwords for multiple

accounts. Keeping track of them on paper or stored on the computer is a huge security risk. By allowing the user to only need to remember a master username and password takes the stress of trying to remember their login information. It also saves them the hassle of having to reset the password.

The user will have our application installed onto their computer. Any changes to the software like updates and bug fixes require a notice of consent. The user will then have to confirm that they are okay with the changes in order to proceed with the update. The user will also be prompted with a notice when there have been multiple attempts to login to their account. Users will have access to our public disclosure through our website and will also be notified when there are updates to the program.

Our program is marketed towards organizations that require the use of multiple accounts that may be shared among the company employees. There is an administrator who has access to all the information stored on the company's database and has the power to reset passwords should the need arise for it. There are employer accounts who can allow employees access to certain shared accounts. The employer can change any of the information on the accounts they have access to. Then there are the employees who only have access to their individual accounts and any accounts shared to them through their employer. Employees can only change the information of their individual accounts and are not permitted to change shared account information.

Our program will prevent unauthorized access to personally identifiable information by restricting access to the information that is theirs and shared with them. It will block access to an

account should the user have several failed login attempts. At this point the user will have to contact the administrator to allow access to the account again.

DESIGN

Design Requirements:

Standard requirements for design are a standard login page to access, create and manage accounts for the user and sub-users. Users should be able to view information on the account that are theirs and under their control. An example of this would be that a new employee access has his own account with a few individual accounts and was allowed access to a shared account.

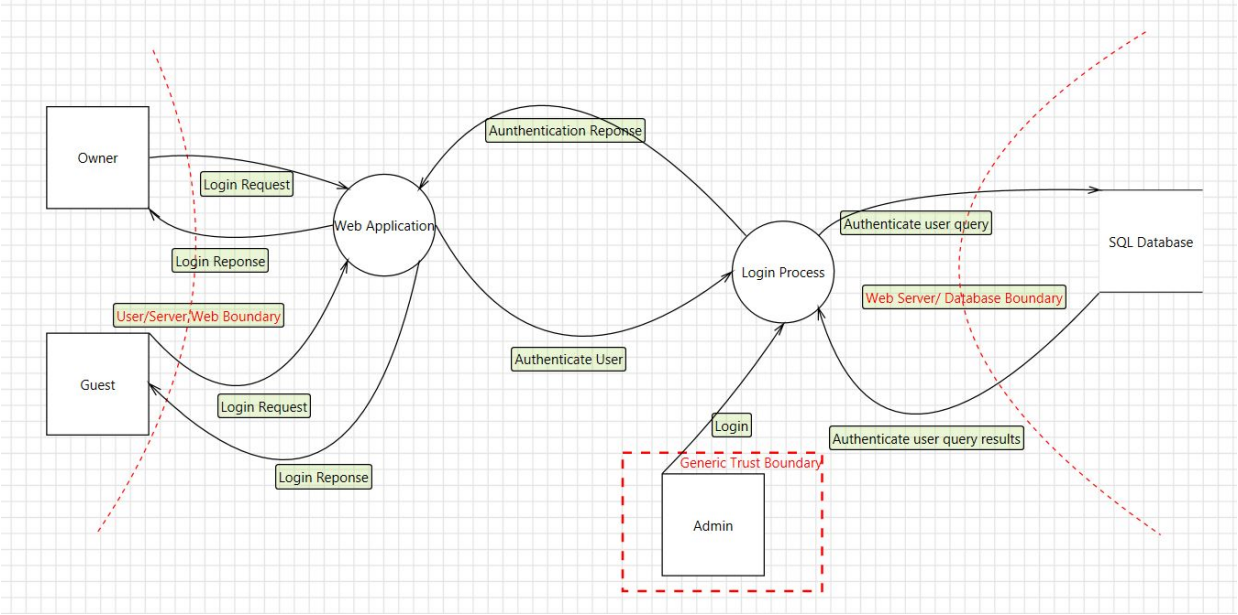
There should be a notification through email and inside the application about any changes to data especially if this data accessed from another computer that has not been authorized access. The notifications should list the relevant information pertaining to where the data was accessed from, when it was accessed, what was changed if anything was changed, and an option to undo the changes. An example would be if a user used a different computer than the one he normally uses. He would then be sent an email notification stating that his account was accessed from a different computer, the time and place it was accessed, and what (if any) changes were made.

There should also be an error page that will prevent a user from being able to login when the user has failed to login several times. An email notification should be sent to the user account that had the failed access attempts to ensure the safety of the account. It will notify the user about the failed attempts to access their account and ask the user to change their password if it was not them attempting to access the account.

Attack Surface Analysis/Reduction:

We should have at least 3 types of privileges level: Level 0 meaning that you are the admin level, and have the most access to all system. Usually for the system developer to access. Level 1 meaning that you are a the manager level or owner of your account, you are able to do actions that relate to your own data. Then finally level 2 which means that you are a guest access, where you have read-only access given to you by the owner of the data. Some vulnerabilities that could affect our security in our program such as open sockets, services, services running by default, services running as system, active web handlers, enabled accounts, enabled accounts in admin group, guest account enabled and weak ACLs.

Threat Modeling:



IMPLEMENTATION

Use Approved Tools:

Compiler/Tool	Minimum Required Version and Switches/Options	Optimal/ Recommended Version and Switches/ Options	Comments
Java SE (JRE/JDK)	Version 7.0	Version 7.0	
Eclipse IDE	Luna (4.4)	Mars (4.5)	
Eclipse Checkstyle Plug-in	Version 6.5.0	n/a	
MySQL Workbench (MySQL Installer)	Version 6.3.4 (Version 5.6.25)	n/a	
WindowBuilder	Version 4.5	n/a	
JGoodies Forms Jar	Version 1.8.0	n/a	This was needed in addition to WindowBuilder to create the GUI.
MySQL Connector Java Jar	Version 5.1.35	n/a	Official Java Database Connectivity driver for MySQL.
Eclipse Checkstyle Plugin	Version 6.5.0	n/a	Unable to install most recent version of Checkstyle

Deprecate Unsafe Functions:

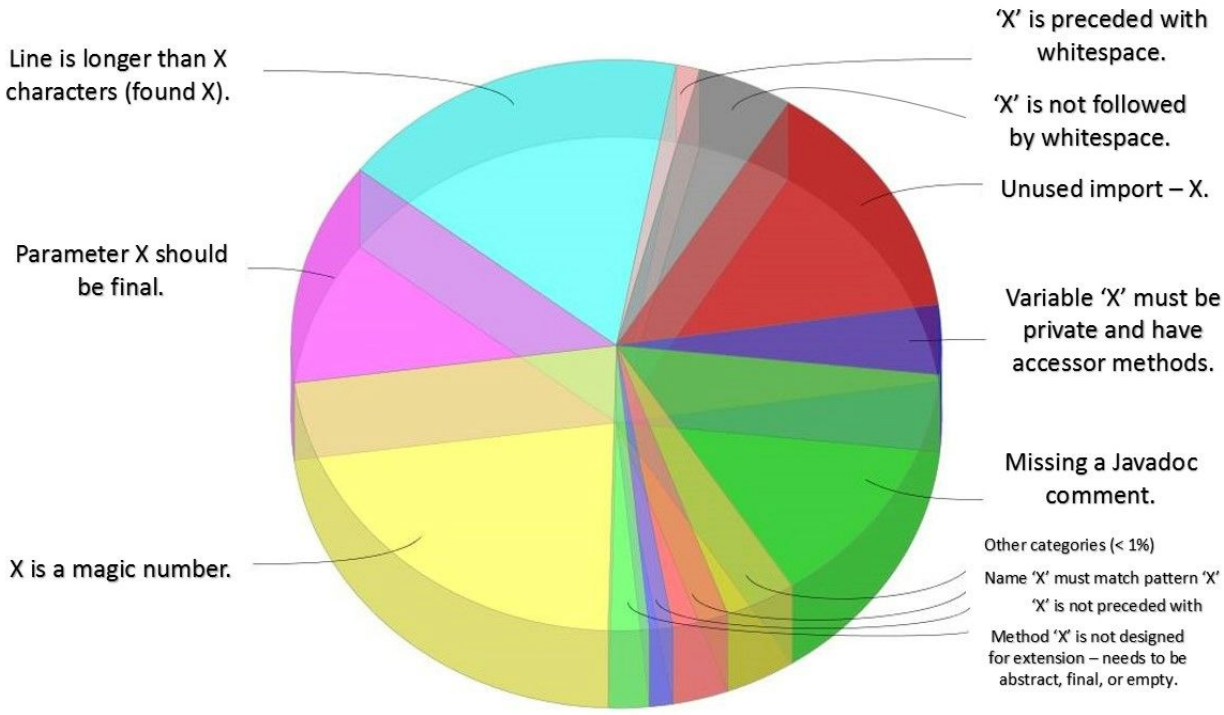
The following is a list of functions that we tried to use or plan to use that have been found to be deprecated (and their safer alternatives, as suggested by API documentations):

Deprecated Function	Alternative
Date(int year, int month, int date)	Calendar.set(year + 1900, month, date)

constructor	
Date(String s)	DateFormat.parse(String s)
JPasswordField.getText()	JPasswordField.getPassword()

Perform Static Analysis:

We chose to use Checkstyle as our static analysis tool. The following are some results we received from our most recent run of the tool:



Checkstyle Violation Type	**Marker Count**
'X' should be on the same line	1
Utility classes should not have a public or default constructor.	1

First sentence should end with a period	1
Expected X tag for ‘X’	2
Missing package-info.java file	2
Array brackets at illegal position	2
‘X’ is not preceded with whitespace	3
‘X’ is preceded with whitespace	3
Method ‘X’ is not designed for extension - needs to be abstract, final or empty	5
Name ‘X’ must match pattern ‘X’	7
Variable ‘X’ must be private and have accessor methods	10
‘X’ is not followed by whitespace	12
Parameter X should be final	32
Unused import - X	35
Missing Javadoc comment	36
Line is longer than X characters (found X)	44

‘X’ is a magic number	57

VERIFICATION

Perform Dynamic Analysis:

For our project we chose to use TestNG for Eclipse. We chose this tool as it is similar to JUnit and it was capable of performing wider tests like unit, functional, end-to-end, and integration. It has other functions like code testing in a multi thread safe, annotations, and flexible configurations for testing. An added bonus is that it is supported by applications like Eclipse, IDEA, and Maven.

We faced the usual challenges when using a new tool. We had to “play around” and experiment with using TestNG. Some of it’s features like automated generation of test methods was a nice feature as compared to JUnit which only give a generic test method. It was interesting experimenting with annotations like not having to extend anything and controlling when a method will be run.

Perform Fuzz Testing:

The following are the testing techniques that we tried and the results that were obtained during the past week. We identified three vulnerabilities and were readily able to patch one of them. The proposed patches for the other two vulnerabilities are presented in the table, as well.

Technique	**Success rate**	**Implications**	**How fix/why secured**
XSS	0%	n/a	Our application is

			currently offline.
SQL Injection - During login	100%	x" or 1=1 # Causes it to skip checking password	Used Prepared Statement to prevent SQL injections. This parametrizes queries properly.
Buffer Overflow - During add new user	100%	Can cause the database to crash if VARCHAR is too large	[Proposed fix] Check for large input. Notify user and/or trim input before passing to DB.
Sensitive Data Exposure - If database is hacked/table data exposed	100%	Passwords stored as plaintext in the DB can allow easy duplication of credentials.	[Proposed fix] Salt and hash login passwords.

Conduct Attack Surface Review:

The following is a review of the tools we have used in the current implementation.

Compiler/Tool	**Version Used in Current Implementation**	**Updates**	**Comments**
Java	Version 7.0	Version 8.0	Stronger algorithms for password-based encryption No new security vulnerabilities reported in currently used version.
Eclipse	Mars (4.5)	n/a	No new versions. No new security

			vulnerabilities reported in currently used version.
Eclipse Checkstyle Plug-in	Version 6.5.0	Version 6.8.0	Latest version is listed as "superficially verified to be compatible with Eclipse Mars" (unfortunately, none of our members have been able to install it). Some bug fixes in v6.8.0. No new security vulnerabilities reported in currently used version.
MySQL Workbench (MySQL Installer)	Version 6.3.4 (Version 5.6.25)	n/a	No new versions. No new security vulnerabilities reported in currently used version.
WindowBuilder	Version 4.5	n/a	No new versions. No new security vulnerabilities reported in currently used version.
JGoodies Forms Jar	Version 1.8.0	Version 1.11.0	Some components used in or compatible with JGoodies Forms have become open-sourced. No new security vulnerabilities reported in currently used version.
MySQL Connector Java Jar	Version 5.1.35	Version 5.1.36	Various bug fixes. No new security vulnerabilities reported in

			currently used version.
TestNG for Eclipse	Version 6.9.4	n/a	No new versions. No new security vulnerabilities reported in currently used version.

RELEASE

<u>Create an Incident Response Plan:</u>

Privacy Escalation Team :

- Escalation Manager (Nicole Isoda)
    - Leads team in handling service issues with customers to gather the overview of the situation and relay it to the individual team members/developers
    - Provides training and guidance to team regarding service and policy/procedure interpretation
- Legal Representative (Nhuc Chau)
    - Handles any privacy-related lawsuits
    - Oversee the work of the team to make sure they are in compliance with company policy'
- Public Relations Representative (Lauren Ogawa)
    - Plan and conducts public relation program to keep public informed of new programs or accomplishments
    - Represents employer during community projects at public, social and business gatherings

- Work with media to advertise products

Emergency contact for Pass Man: lvl7.pwmanagement@gmail.com

Incident response Procedures:

1. Customer submits report with information containing error message, things they were trying to accomplish and a few steps of how they came to the error.
2. Escalation manager sorts the reports and keep in contact with the customer in case more information is required.
3. Team works on analyzing the situation and determines the critical level and pushes it to developers to fix
4. After finding a resolution, escalation manager documents the process in case similar problems arise

Conduct Final Security Review:

Summary

The Pass Man is an application that allows the user to manage their login information to different websites, just by remembering one login. All the user needs is a master login username and password to access all of their other saved login information that are stored in a secure database. From there they are able to add and delete as much of their entries as they desire!

Assets and Security Goals

1. The information stored by the application should not be given to any unintended parties. This means that access to a user's account information should only be given to the user who own's the confidential information.
2. The application should be used to store only the user's personal account information.
3. The application is a benefit and should be secure from basic tampering. Else, the information may be modified and compromised.

Adversaries and Threats

1. Competing companies may tamper with the product to cause it malfunction and cause information to be relayed insecurely. Causing damage to not only the user's information but also Level 7's reputation and application.
2. Persons with malicious intent against the user may attempt to hack into the application or its database to obtain PII. This could result in the attacker being able to access the accounts stored by the user.

Potential Weaknesses/Exceptions

1. An attacker that is able to gain access to the application's database will have full access to all stored information. The database must therefore be secured and PII in the database may need to be altered in the future to ensure an attacker cannot use the information.

Final Grade

After reviewing the application, security and privacy goals, potential threats, and any potential weaknesses/security exceptions, the team has decided to give Pass Man a grade of ***Passed FSR with Exceptions***. There is currently a security issue in the application that does not immediately hinder functionality of the app, but this issue should be addressed and patched in the

next release of the application. All other issues found during testing and FSR have been patched appropriately.

Certify Release and Archive:

The source code, wiki, and other necessary documentation for Pass Man (Release 1.0.0) are available here: https://github.com/nisoda/level-7-pass-man.git