Software Version Description

for

TheraStation

Version 1.0 approved

**Prepared by Gabrielle Hoff, PhucSang Le,
Franklin Vega, Joshua Wilkes**

SEM Group 3

26 October 2018

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
TheraStationSVD	26 October 2018	Initial Version	1.0

1. Introduction

1.1 System Overview

TheraStation is a website that allows users to find and schedule appointments with therapists and other mental health professionals. The physicians will be able to receive messages and keep notes on the patients/users.

1.2 Version Overview

The Sprint 1 release of TheraStation consists of a login, account creation, user profiles, a basic listing

1.3 Team Assignments

- Gabrielle Hoff: Backend Developer
- PhucSang Le: Operations
- Franklin Vega: Scrum Master and Floating Developer
- Joshua Wilkes: Frontend Developer

1.4 Document Conventions

1.4.1 Typographical Conventions

Main sections are given by a single number followed by the title of the section, with each subsection following the same convention, this is done by adding a period followed by adding an additional number

1.4.2 Contextual Conventions

Across the entirety of the VSD, any occurrence of the word “our,” or any other word that implies grouping or collective ownership, refers to the TheraStation development team; the Therastation team consists of Gabrielle Hoff, Jimmy Le, Joshua Wilkes, and Franklin Vega

1.5 References

Home - Mongo Express Angular Node. (n.d.). Retrieved from http://mean.io/

2. Inventory of Materials

2.1 Documents Released

2.1.1 Included Documents

2.1.1.1 TheraStationSVD.pdf (this document)
2.1.1.2 README.md

2.2 Executable Media Released

Due to this being a web browser application, no executable files will be included in this software system.

2.3 Software Projects\Assemblies Released

2.4.1 Server

- An assembly for the node server application

2.4.2 Src

- An assembly that manages the database and storing information.

2.4 Test Projects\Scripts Released

2.4.2 '.spec.ts' Test Files

- Tests components in the source (src) folder
- Auto generated from Angular CLI

3. Current Design

3.1 Current Software Layers

TheraStation utilizes a MEAN Stack architecture. MEAN is an acronym which stands for: MongoDB, Express, Angular, Node.js.

3.1.1 MongoDB

- NoSQL database used for storing data

3.1.2 Express

- Web application framework that runs on Node.js

3.1.2 Angular

- A Model-View-Controller framework that runs in browser-based javascript engines

3.1.3 Node.js

- The database that is being used is a Microsoft Azure SQL database. The database holds the user's account information, including their log-in.

3.2 Major Software Interfaces

3.2.1 User Interface Layer

- The user will enter their account information to log in and they are given the choice to either schedule a tutor or study session. This is all completed using Angular, which is accessed via web browser.

3.2.2 Logic Layer

- The user will enter their account information and this will request access from the database to determine whether the account is valid or invalid. If the account is valid, the user will be logged in.

3.2.3 Data Access Layer

- The database that is being used is a MongoDB NoSQL database. The database holds the user's account information, including their log-in.

3.3 Incorporating Design Patterns

For the design pattern, TheraStation utilizes a MEAN Stack architecture. MEAN is an acronym which stands for: MongoDB, Express, Angular, Node.js.

3.4 Data Schema

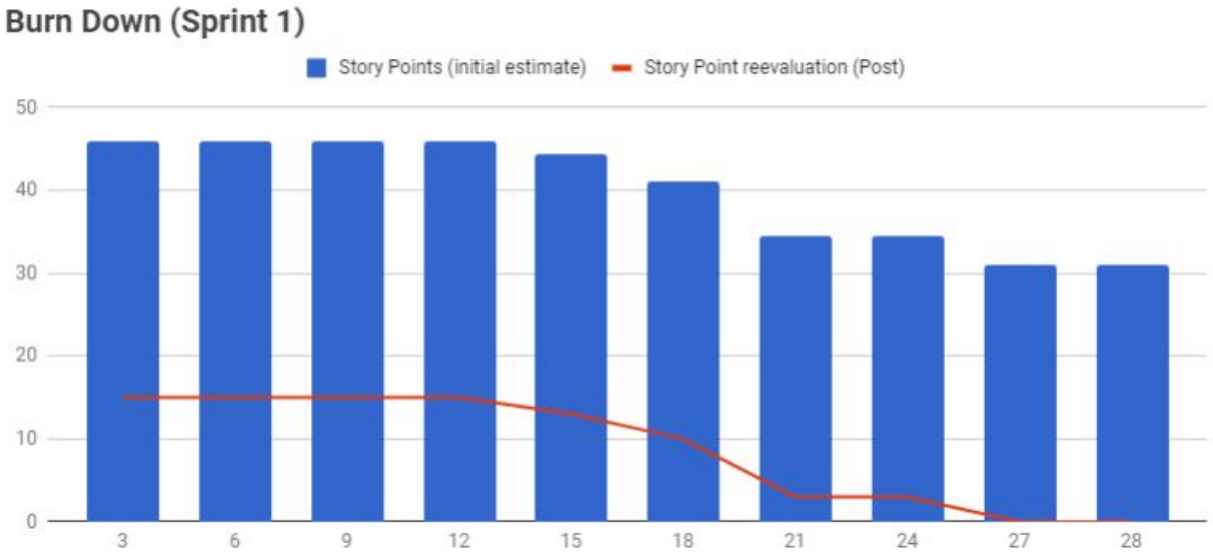
TheraStation utilizes MongoDB, a NoSQL database. Rather than using commands and tables like those used in SQL, MongoDB utilizes JSON like objects.

3.5 Violations to Design Principles

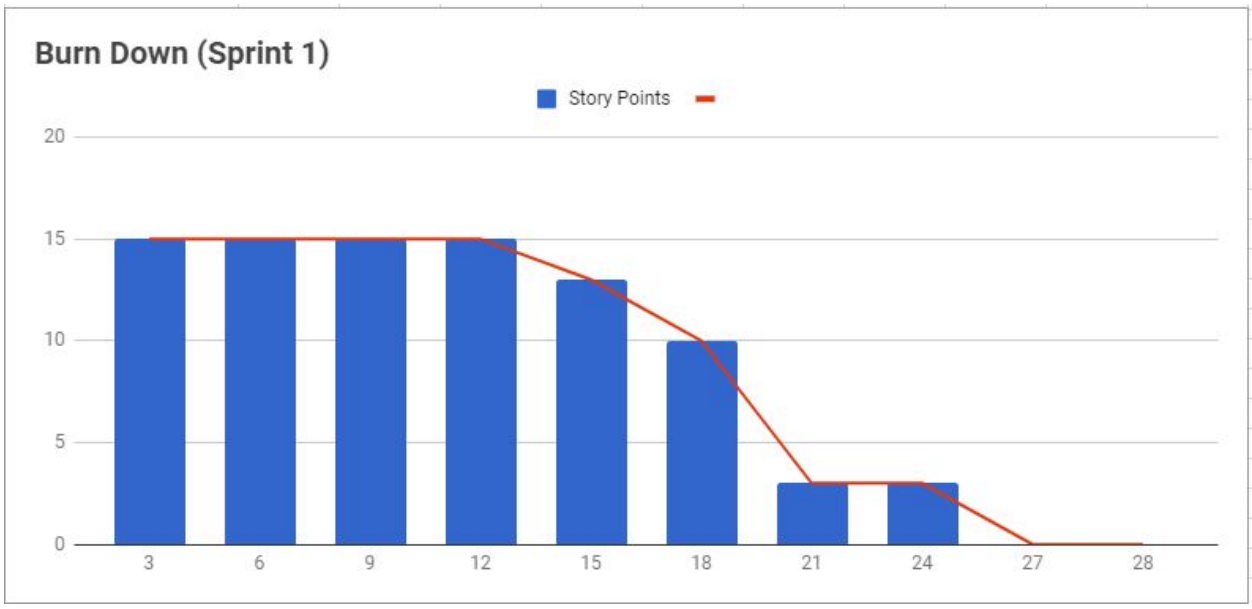
There are no currently known violations to any of the SOLID design principles.

4. Tracking and Monitoring

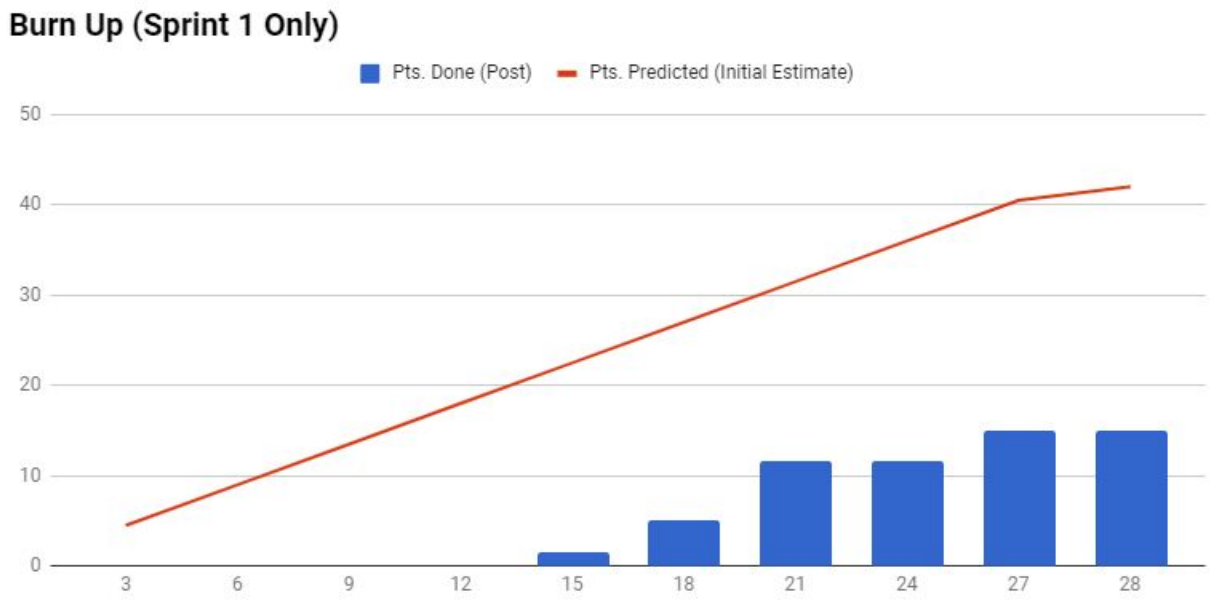
4.1 Burn Up Chart



After a post reevaluation of the story points completed this sprint, the burndown would look more like the following:



4.2 Burn Down Chart



4.3 Current Velocity

For Sprint 1, we predicted a velocity of 46 story points total using planning poker. These 46 story points are spread across 9 separate tasks (as seen on the Trello board). However, 6 of the 9 tasks overlap both the Sprint 1 & 2 goals. The first 3, non-overlapping tasks were of higher priority and were derived from the goal of Sprint 1. It is also important to note that due to this being the initial sprint, some of these computed values may be a bit skewed or inaccurate.

4.4 Value Added

Adding all of the initial 9 tasks under Sprint 1 resulted in an initial value of 46 estimated story points. as previously stated in section 4.3, due to this being the initial sprint, some of these computed values may be a bit skewed or inaccurate. If the 3 core tasks were only counted, there would only be 15 story points total, so these tasks may have been underestimated during planning poker. These were generally assumed to be around 5 points each, however in reality, these may have had much more story points.

5. Remaining Work

5.1 Remaining Software Features

5.1.1 Advanced Search Features

- Search by review

5.1.2 Feedback Features

- Create a review
- Display reviews

5.1.3 Messaging Features

- Send/reply to messages
- Read/receive messages

5.2 Known Issues and Bugs

There are currently no known issues or bugs with the current build of TheraStation

Appendix A: Glossary

1. MEAN : An acronym which stands for: MongoDB, Express, Angular, Node.js.
2. SOLID design principles: Design principle acronym that stands for: Single responsibility principle, Open/close principle, Liskov substitution principle, Interface segregation principle, Dependency inversion principle.
3. Single responsibility principle: States that a class should only have one reason to change
4. Open/close principle: States that software entities should be open for extension, but closed for modification.
5. Liskov substitution principle: States that subtypes must be substitutable for their base types.
6. Interface segregation principle: States that clients should not be forced to depend on methods that they do not use.
7. Dependency inversion principle: States that high level models should not depend on low level models.
8. JSON: Data format that frequently accompanies javascript. \
9. Mocha: JavaScript framework that runs on node.js and in the browser

The principle definitions were found at https://deviq.com/solid/

Appendix B: GitHub Information

- GitHub URL:
  - https://github.com/UWF-HMCSE-CS/SEM2018_Group3
- Trello Board URL:
  - https://trello.com/b/fI6nVUHd/sem-group-3