Conversational Agent - Project Document

The University of British Columbia - Okanagan

COSC 310 - Software Engineering

Professor: Dr. Narayan Apurva

April 4, 2020

Jaden Balogh (38360673)
Jonathan Gresl (29164977)
Michael Crouse (19271824)
Shamus Boulianne (78559770)

Table of Contents

Agent Role: 2
Conversation Topics: 2
Project Description: 2
SDLC Choice: 3
SDLC Rational: 3
Potential API Features: 3
SDLC Phases: 4
Work Break Down Structure (WBS): 7
Gantt Chart: 9
Data Flow Diagrams: 11
Sample Output: 13
Commits by User: 16
Task Hour Summary: 17
Evaluation: 19

Agent Role:

The “agent” will act as the US president, Donald Trump.

Conversation Topics:

- A young child that asks the US president various questions about his job.
- A discussion with the US president about politics and campaigns.

Project Description:

Our project aims to capture the spirit of Donald Trump in a chatbot that allows users a comedic interaction with the US president who uses his typical catchphrases and funny phrasings to talk to you. The user acts as a person who is curious about the president, his job, and his opinions on various things. We used a React.js solution for this project to allow for a web-based interface for easy use. The application is hosted on an external web server accessible at http://50.98.99.115:5000/. A video describing the project can be found on YouTube at https://youtu.be/hvAclW_ebs8. The project code and supporting documentation can be found on GitHub at https://github.com/jgresl/trump-bot.

SDLC Choice:

Our group used a modified scrum where we utilize an online communication platform (Slack) to communicate on a regular basis in place of daily meetings. Our sprints are 7 days long and we meet weekly – for 2 hours at the end of each sprint – to review the past sprint and prepare for the next one. We also utilize this time to collaboratively act as the product manager by re-prioritizing the items in the product backlog.

SDLC Rational:

With our group being composed of students, we wanted to organize ourselves in a democratic model – rather than hierarchical. Although the requirements of the final product are clear, the overall scope is not because we are learning what needs to be done as we go. That is, we weren't able to identify every task that needed to be done at the start of the project.

The scrum style is most suitable for our group project because:

- Individuals can sign up for work of their own choosing
- Any team member can add, delete change the sprint backlog
- Remaining work can be updated as more becomes known

Potential API Features:

The majority of our chatbot's functionality could be extracted to an API for general use. Some potential API features are listed below as functions that would be accessible to external users.

- getResponse(inputText): returns a response generated from the response tree based on some user input string. Basic functionality of the chatbot.
- getResponseTree(): returns the root of the response tree. Useful for debugging or checking specific values in the tree.
- addResponse(isQuestion, isPositive, matchWords, responseString): adds a response to the response tree with the given parameters. This allows the user to customize and improve the possible responses.
- removeResponse(isQuestion, isPositive, matchWords): finds and removes the response associated with the given parameters. This allows the user to customize and improve the possible responses.
- processText(inputText): returns a list of strings from running our natural language processing algorithm on a given input string. Extracts the language functionality from the chatbot algorithm for general use.

SDLC Phases:

- **Phase 1 (A2 - Sprint Backlog 1)**
  - **Requirements**
    - Research chatbot algorithms
  - **Code**
    - Create base application
  - **Test**
    - Run base application
  - **Documentation**
    - Setup environment
      - Create WBS / GANTT Chart
      - Setup Github repository
      - Create README template
    - Write wiki page for react.js

- **Phase 2 (A2 - Sprint Backlog 2)**
  - **Requirements**
    - Setup web server to host production site
    - Sample conversation 1
    - Initial text file for output tree
  - **Design**
    - Design high-level processing algorithm
  - **Code**
    - Implement text pre-processing
    - Implement text input processing
  - **Test**
    - Test web hosting
  - **Documentation**
    - Add SDLC rational to project document

- **Phase 3 (A3 - Sprint Backlog 3)**
  - **Requirements**
    - Align requirements with deliverables
  - **Code**
    - Text mapping algorithm
    - Basic text input and output UI
    - Output generation
  - **Test**
    - Test and prepare for 30 turn conversation

        - Test and review code changes
    - **Documentation**
        - Finalize project document
            - Add description to document
            - Add SDLC phases / tasks to document
            - Add final WBS / GANTT to document
            - Add list of limitations to document
        - Populate sample output file in Github
        - Add Github commits by user
        - Finalize README file in GitHub repository

- **Phase 4 (Project - Sprint Backlog 1)**
    - **Requirements**
        - Learn natural NLP functionality and use
        - Research neural net/machine learning
    - **Code**
        - User input NLP pre-processing
        - Add weighted logic to selection from response list
        - Save history of past responses
        - Start advanced GUI

- **Phase 5 (Project - Sprint Backlog 2)**
    - **Requirements**
        - Learn about React states and components
        - Learn about slack integration
    - **Code**
        - Update isQuestion method
        - Add server-side express functionality
        - Add scroll bar to chat UI
        - Finalize UI message styling

- **Phase 6 (Project - Sprint Backlog 3)**
    - **Design**
        - Design advanced response mapping
    - **Code**
        - Implement Slack chat-bot integration
        - Connect GUI to message history
        - Add auto-scroll function to GUI
        - Implement natural functions server-side
        - Expand tree node input functionality

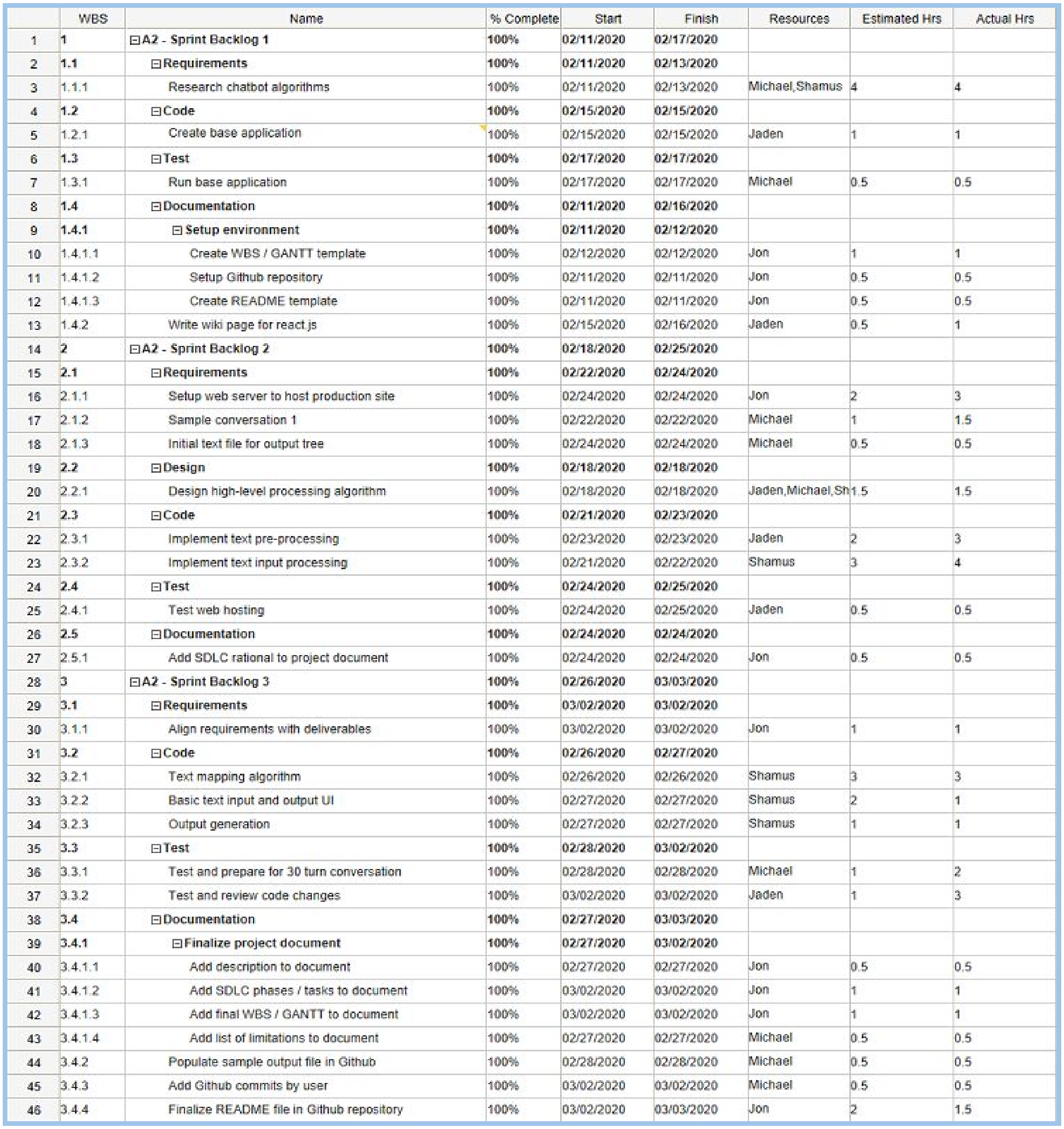
    - Implement component-based GUI
    - Integrate GUI with server-side functionality
  - **Test**
    - Test and troubleshoot new master branch
    - Test natural functions server-side

- **Phase 7 (Project - Sprint Backlog 4)**
  - **Design**
    - Add extra topic to agent repertoire
    - Add 5 non-topic related responses
  - **Code**
    - Implement feature to handle spelling mistakes
    - Add synonyms to language processing
  - **Test**
    - Test pre-processing for bugs
  - **Documentation**
    - Add list of 5 features that can be used as API
    - Walk through of each feature in README file
    - Provide level 0 DFD
    - Provide level 1 DFD
  - **Presentation**
    - Begin writing script

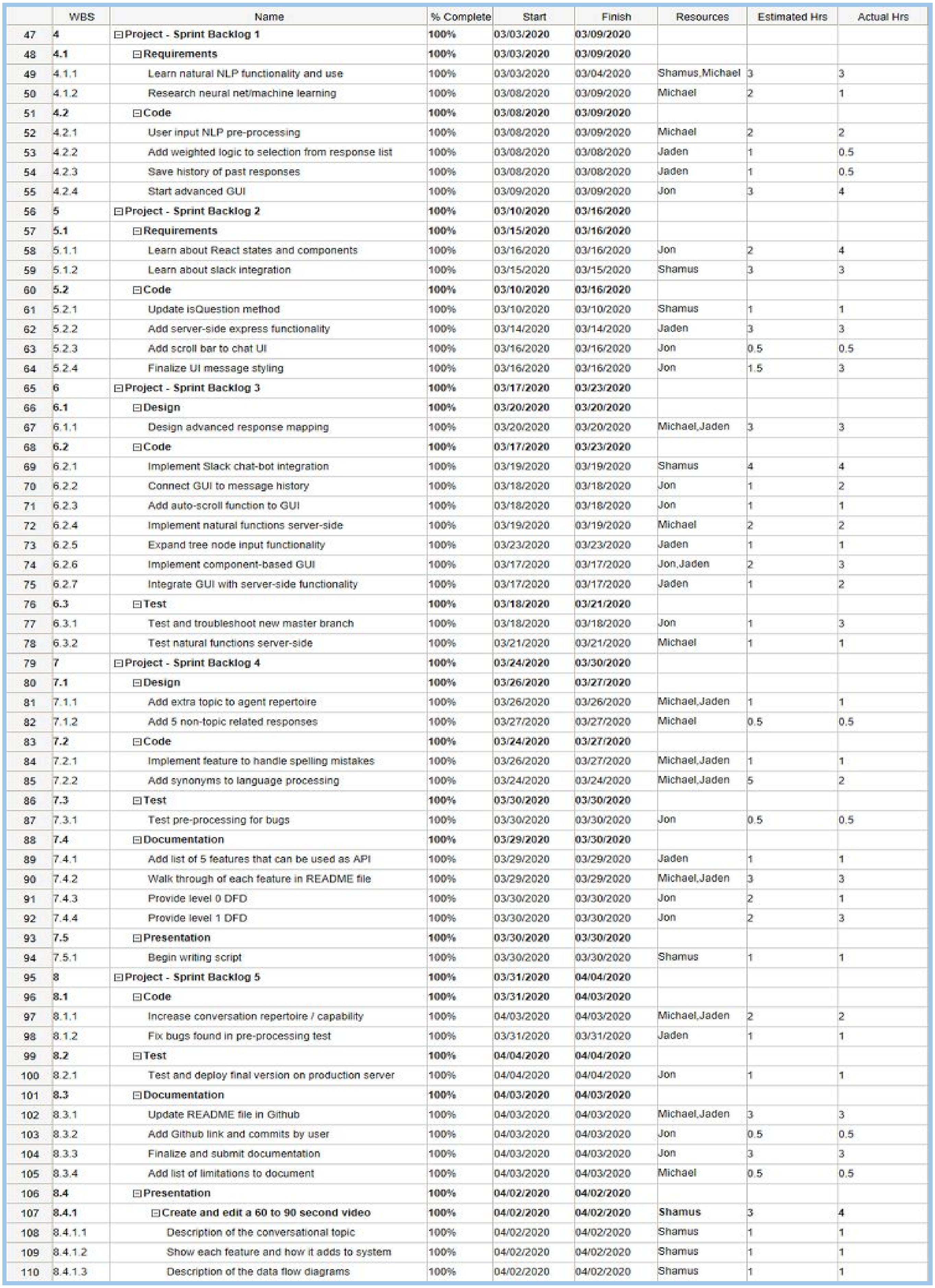
- **Phase 8 (Project - Sprint Backlog 5)**
  - **Code**
    - Increase conversation repertoire / capability
    - Fix bugs found in pre-processing test
  - **Test**
    - Test and deploy final version on production server
  - **Documentation**
    - Update README file in Github
    - Add Github link and commits by user
    - Finalize and submit documentation
    - Add list of limitations to document
  - **Presentation**
    - Create and edit a 60 to 90 second video
      - Description of the conversational topic
      - Show each feature and how it adds to system
      - Description of the data flow diagrams

Work Break Down Structure (WBS):

Our Work Breakdown Structure follows a scrum format with our tasks split into sprints. At the start of each sprint, our team had a meeting in which we moved the highest priority features into the sprint backlog, which were then subdivided into individual smaller tasks. Throughout the week, team members assigned themselves to tasks and updated the WBS with their progress.



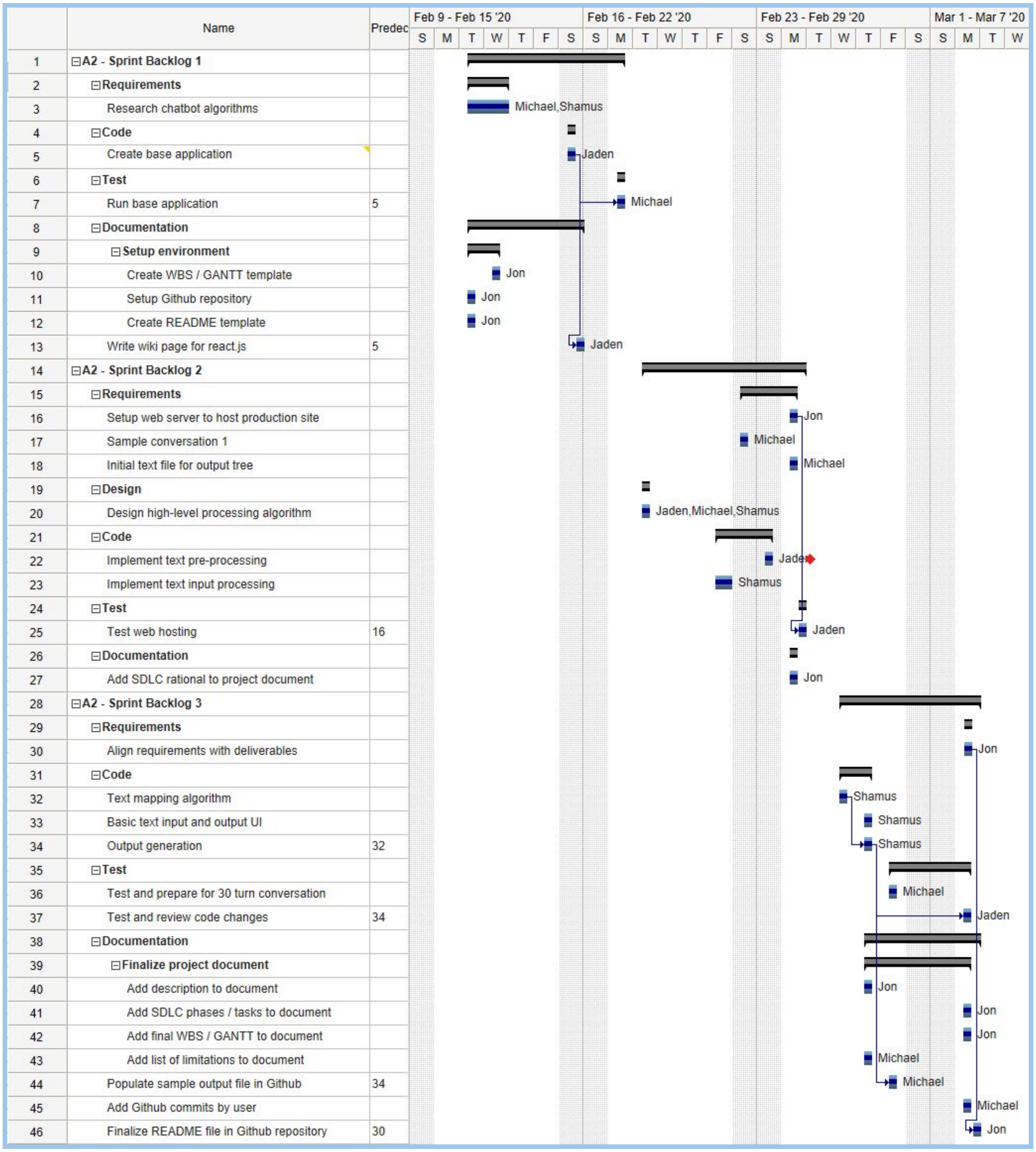
	WBS	Name	% Complete	Start	Finish	Resources	Estimated Hrs	Actual Hrs
1	**1**	**⊟A2 - Sprint Backlog 1**	**100%**	**02/11/2020**	**02/17/2020**			
2	**1.1**	**⊟Requirements**	**100%**	**02/11/2020**	**02/13/2020**			
3	1.1.1	Research chatbot algorithms	100%	02/11/2020	02/13/2020	Michael,Shamus	4	4
4	**1.2**	**⊟Code**	**100%**	**02/15/2020**	**02/15/2020**			
5	1.2.1	Create base application	100%	02/15/2020	02/15/2020	Jaden	1	1
6	**1.3**	**⊟Test**	**100%**	**02/17/2020**	**02/17/2020**			
7	1.3.1	Run base application	100%	02/17/2020	02/17/2020	Michael	0.5	0.5
8	**1.4**	**⊟Documentation**	**100%**	**02/11/2020**	**02/16/2020**			
9	**1.4.1**	**⊟Setup environment**	**100%**	**02/11/2020**	**02/12/2020**			
10	1.4.1.1	Create WBS / GANTT template	100%	02/12/2020	02/12/2020	Jon	1	1
11	1.4.1.2	Setup Github repository	100%	02/11/2020	02/11/2020	Jon	0.5	0.5
12	1.4.1.3	Create README template	100%	02/11/2020	02/11/2020	Jon	0.5	0.5
13	1.4.2	Write wiki page for react.js	100%	02/15/2020	02/16/2020	Jaden	0.5	1
14	**2**	**⊟A2 - Sprint Backlog 2**	**100%**	**02/18/2020**	**02/25/2020**			
15	**2.1**	**⊟Requirements**	**100%**	**02/22/2020**	**02/24/2020**			
16	2.1.1	Setup web server to host production site	100%	02/24/2020	02/24/2020	Jon	2	3
17	2.1.2	Sample conversation 1	100%	02/22/2020	02/22/2020	Michael	1	1.5
18	2.1.3	Initial text file for output tree	100%	02/24/2020	02/24/2020	Michael	0.5	0.5
19	**2.2**	**⊟Design**	**100%**	**02/18/2020**	**02/18/2020**			
20	2.2.1	Design high-level processing algorithm	100%	02/18/2020	02/18/2020	Jaden,Michael,Sh	1.5	1.5
21	**2.3**	**⊟Code**	**100%**	**02/21/2020**	**02/23/2020**			
22	2.3.1	Implement text pre-processing	100%	02/23/2020	02/23/2020	Jaden	2	3
23	2.3.2	Implement text input processing	100%	02/21/2020	02/22/2020	Shamus	3	4
24	**2.4**	**⊟Test**	**100%**	**02/24/2020**	**02/25/2020**			
25	2.4.1	Test web hosting	100%	02/24/2020	02/25/2020	Jaden	0.5	0.5
26	**2.5**	**⊟Documentation**	**100%**	**02/24/2020**	**02/24/2020**			
27	2.5.1	Add SDLC rational to project document	100%	02/24/2020	02/24/2020	Jon	0.5	0.5
28	**3**	**⊟A2 - Sprint Backlog 3**	**100%**	**02/26/2020**	**03/03/2020**			
29	**3.1**	**⊟Requirements**	**100%**	**03/02/2020**	**03/02/2020**			
30	3.1.1	Align requirements with deliverables	100%	03/02/2020	03/02/2020	Jon	1	1
31	**3.2**	**⊟Code**	**100%**	**02/26/2020**	**02/27/2020**			
32	3.2.1	Text mapping algorithm	100%	02/26/2020	02/26/2020	Shamus	3	3
33	3.2.2	Basic text input and output UI	100%	02/27/2020	02/27/2020	Shamus	2	1
34	3.2.3	Output generation	100%	02/27/2020	02/27/2020	Shamus	1	1
35	**3.3**	**⊟Test**	**100%**	**02/28/2020**	**03/02/2020**			
36	3.3.1	Test and prepare for 30 turn conversation	100%	02/28/2020	02/28/2020	Michael	1	2
37	3.3.2	Test and review code changes	100%	03/02/2020	03/02/2020	Jaden	1	3
38	**3.4**	**⊟Documentation**	**100%**	**02/27/2020**	**03/03/2020**			
39	**3.4.1**	**⊟Finalize project document**	**100%**	**02/27/2020**	**03/02/2020**			
40	3.4.1.1	Add description to document	100%	02/27/2020	02/27/2020	Jon	0.5	0.5
41	3.4.1.2	Add SDLC phases / tasks to document	100%	03/02/2020	03/02/2020	Jon	1	1
42	3.4.1.3	Add final WBS / GANTT to document	100%	03/02/2020	03/02/2020	Jon	1	1
43	3.4.1.4	Add list of limitations to document	100%	02/27/2020	02/27/2020	Michael	0.5	0.5
44	3.4.2	Populate sample output file in Github	100%	02/28/2020	02/28/2020	Michael	0.5	0.5
45	3.4.3	Add Github commits by user	100%	03/02/2020	03/02/2020	Michael	0.5	0.5
46	3.4.4	Finalize README file in Github repository	100%	03/02/2020	03/03/2020	Jon	2	1.5



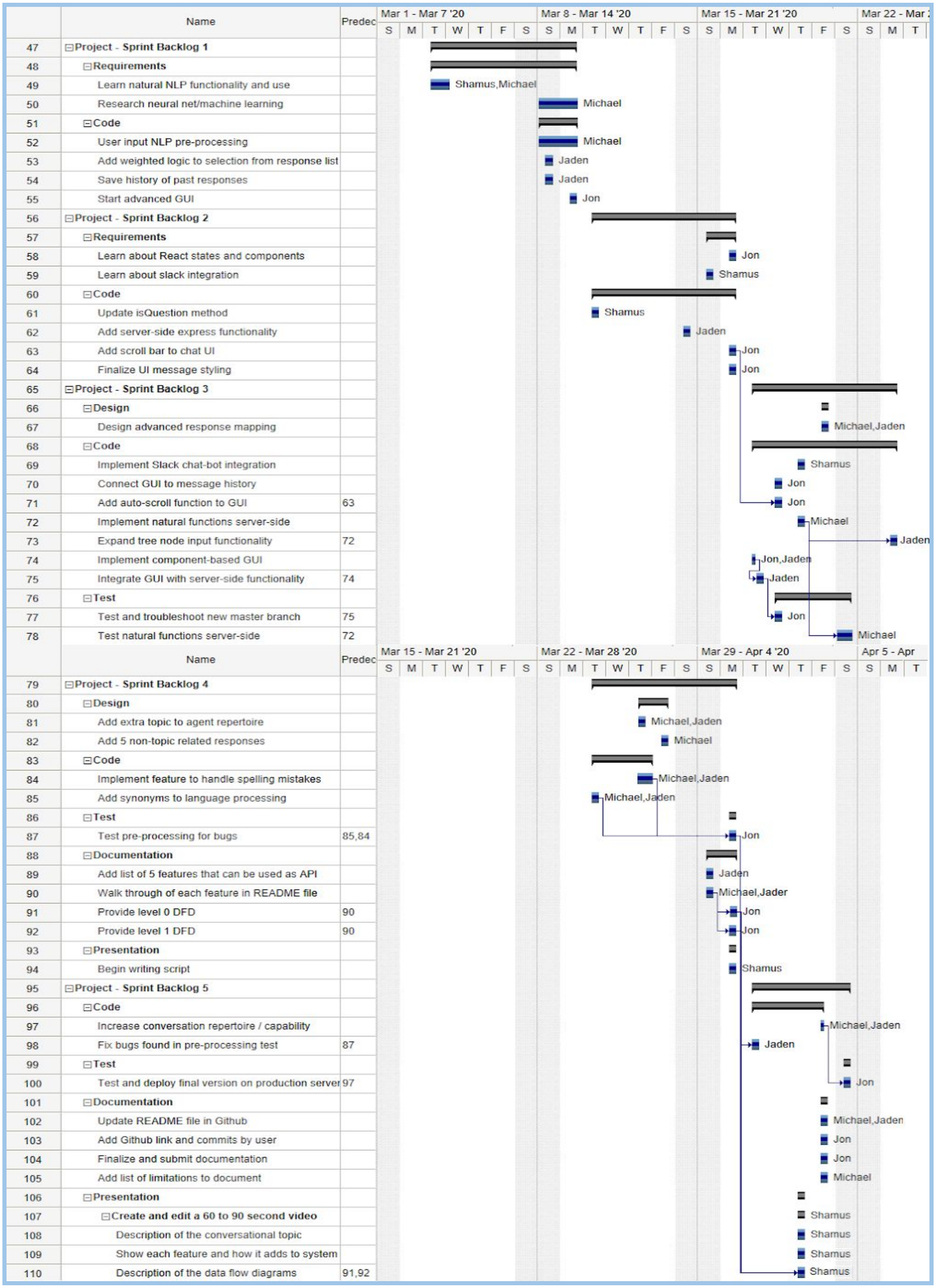
	WBS	Name	% Complete	Start	Finish	Resources	Estimated Hrs	Actual Hrs
47	**4**	**⊟Project - Sprint Backlog 1**	**100%**	**03/03/2020**	**03/09/2020**			
48	**4.1**	**⊟Requirements**	**100%**	**03/03/2020**	**03/09/2020**			
49	4.1.1	Learn natural NLP functionality and use	100%	03/03/2020	03/04/2020	Shamus,Michael	3	3
50	4.1.2	Research neural net/machine learning	100%	03/08/2020	03/09/2020	Michael	2	1
51	**4.2**	**⊟Code**	**100%**	**03/08/2020**	**03/09/2020**			
52	4.2.1	User input NLP pre-processing	100%	03/08/2020	03/09/2020	Michael	2	2
53	4.2.2	Add weighted logic to selection from response list	100%	03/08/2020	03/08/2020	Jaden	1	0.5
54	4.2.3	Save history of past responses	100%	03/08/2020	03/08/2020	Jaden	1	0.5
55	4.2.4	Start advanced GUI	100%	03/09/2020	03/09/2020	Jon	3	4
56	**5**	**⊟Project - Sprint Backlog 2**	**100%**	**03/10/2020**	**03/16/2020**			
57	**5.1**	**⊟Requirements**	**100%**	**03/15/2020**	**03/16/2020**			
58	5.1.1	Learn about React states and components	100%	03/16/2020	03/16/2020	Jon	2	4
59	5.1.2	Learn about slack integration	100%	03/15/2020	03/15/2020	Shamus	3	3
60	**5.2**	**⊟Code**	**100%**	**03/10/2020**	**03/16/2020**			
61	5.2.1	Update isQuestion method	100%	03/10/2020	03/10/2020	Shamus	1	1
62	5.2.2	Add server-side express functionality	100%	03/14/2020	03/14/2020	Jaden	3	3
63	5.2.3	Add scroll bar to chat UI	100%	03/16/2020	03/16/2020	Jon	0.5	0.5
64	5.2.4	Finalize UI message styling	100%	03/16/2020	03/16/2020	Jon	1.5	3
65	**6**	**⊟Project - Sprint Backlog 3**	**100%**	**03/17/2020**	**03/23/2020**			
66	**6.1**	**⊟Design**	**100%**	**03/20/2020**	**03/20/2020**			
67	6.1.1	Design advanced response mapping	100%	03/20/2020	03/20/2020	Michael,Jaden	3	3
68	**6.2**	**⊟Code**	**100%**	**03/17/2020**	**03/23/2020**			
69	6.2.1	Implement Slack chat-bot integration	100%	03/19/2020	03/19/2020	Shamus	4	4
70	6.2.2	Connect GUI to message history	100%	03/18/2020	03/18/2020	Jon	1	2
71	6.2.3	Add auto-scroll function to GUI	100%	03/18/2020	03/18/2020	Jon	1	1
72	6.2.4	Implement natural functions server-side	100%	03/19/2020	03/19/2020	Michael	2	2
73	6.2.5	Expand tree node input functionality	100%	03/23/2020	03/23/2020	Jaden	1	1
74	6.2.6	Implement component-based GUI	100%	03/17/2020	03/17/2020	Jon,Jaden	2	3
75	6.2.7	Integrate GUI with server-side functionality	100%	03/17/2020	03/17/2020	Jaden	1	2
76	**6.3**	**⊟Test**	**100%**	**03/18/2020**	**03/21/2020**			
77	6.3.1	Test and troubleshoot new master branch	100%	03/18/2020	03/18/2020	Jon	1	3
78	6.3.2	Test natural functions server-side	100%	03/21/2020	03/21/2020	Michael	1	1
79	**7**	**⊟Project - Sprint Backlog 4**	**100%**	**03/24/2020**	**03/30/2020**			
80	**7.1**	**⊟Design**	**100%**	**03/26/2020**	**03/27/2020**			
81	7.1.1	Add extra topic to agent repertoire	100%	03/26/2020	03/26/2020	Michael,Jaden	1	1
82	7.1.2	Add 5 non-topic related responses	100%	03/27/2020	03/27/2020	Michael	0.5	0.5
83	**7.2**	**⊟Code**	**100%**	**03/24/2020**	**03/27/2020**			
84	7.2.1	Implement feature to handle spelling mistakes	100%	03/26/2020	03/27/2020	Michael,Jaden	1	1
85	7.2.2	Add synonyms to language processing	100%	03/24/2020	03/24/2020	Michael,Jaden	5	2
86	**7.3**	**⊟Test**	**100%**	**03/30/2020**	**03/30/2020**			
87	7.3.1	Test pre-processing for bugs	100%	03/30/2020	03/30/2020	Jon	0.5	0.5
88	**7.4**	**⊟Documentation**	**100%**	**03/29/2020**	**03/30/2020**			
89	7.4.1	Add list of 5 features that can be used as API	100%	03/29/2020	03/29/2020	Jaden	1	1
90	7.4.2	Walk through of each feature in README file	100%	03/29/2020	03/29/2020	Michael,Jaden	3	3
91	7.4.3	Provide level 0 DFD	100%	03/30/2020	03/30/2020	Jon	2	1
92	7.4.4	Provide level 1 DFD	100%	03/30/2020	03/30/2020	Jon	2	3
93	**7.5**	**⊟Presentation**	**100%**	**03/30/2020**	**03/30/2020**			
94	7.5.1	Begin writing script	100%	03/30/2020	03/30/2020	Shamus	1	1
95	**8**	**⊟Project - Sprint Backlog 5**	**100%**	**03/31/2020**	**04/04/2020**			
96	**8.1**	**⊟Code**	**100%**	**03/31/2020**	**04/03/2020**			
97	8.1.1	Increase conversation repertoire / capability	100%	04/03/2020	04/03/2020	Michael,Jaden	2	2
98	8.1.2	Fix bugs found in pre-processing test	100%	03/31/2020	03/31/2020	Jaden	1	1
99	**8.2**	**⊟Test**	**100%**	**04/04/2020**	**04/04/2020**			
100	8.2.1	Test and deploy final version on production server	100%	04/04/2020	04/04/2020	Jon	1	1
101	**8.3**	**⊟Documentation**	**100%**	**04/03/2020**	**04/03/2020**			
102	8.3.1	Update README file in Github	100%	04/03/2020	04/03/2020	Michael,Jaden	3	3
103	8.3.2	Add Github link and commits by user	100%	04/03/2020	04/03/2020	Jon	0.5	0.5
104	8.3.3	Finalize and submit documentation	100%	04/03/2020	04/03/2020	Jon	3	3
105	8.3.4	Add list of limitations to document	100%	04/03/2020	04/03/2020	Michael	0.5	0.5
106	**8.4**	**⊟Presentation**	**100%**	**04/02/2020**	**04/02/2020**			
107	**8.4.1**	**⊟Create and edit a 60 to 90 second video**	**100%**	**04/02/2020**	**04/02/2020**	**Shamus**	**3**	**4**
108	8.4.1.1	Description of the conversational topic	100%	04/02/2020	04/02/2020	Shamus	1	1
109	8.4.1.2	Show each feature and how it adds to system	100%	04/02/2020	04/02/2020	Shamus	1	1
110	8.4.1.3	Description of the data flow diagrams	100%	04/02/2020	04/02/2020	Shamus	1	1

Gantt Chart:

Our Gantt chart, generated directly from our WBS, shows the progression of work completed through each sprint (shown as the largest gray bars in the chart). As shown in the chart, almost all tasks were short enough to only require one day or less of work, spread out over the week but more concentrated on weekends. Each sprint was 7 days and tended to shift from a focus on requirements, to coding, to documentation and testing.

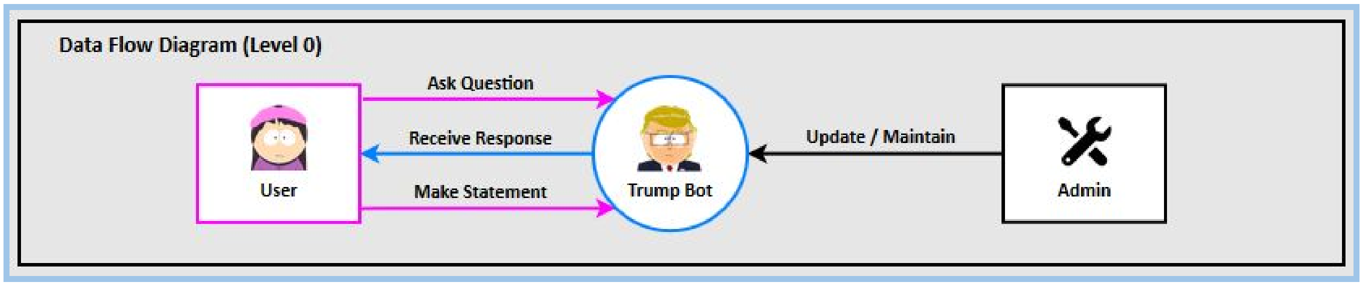


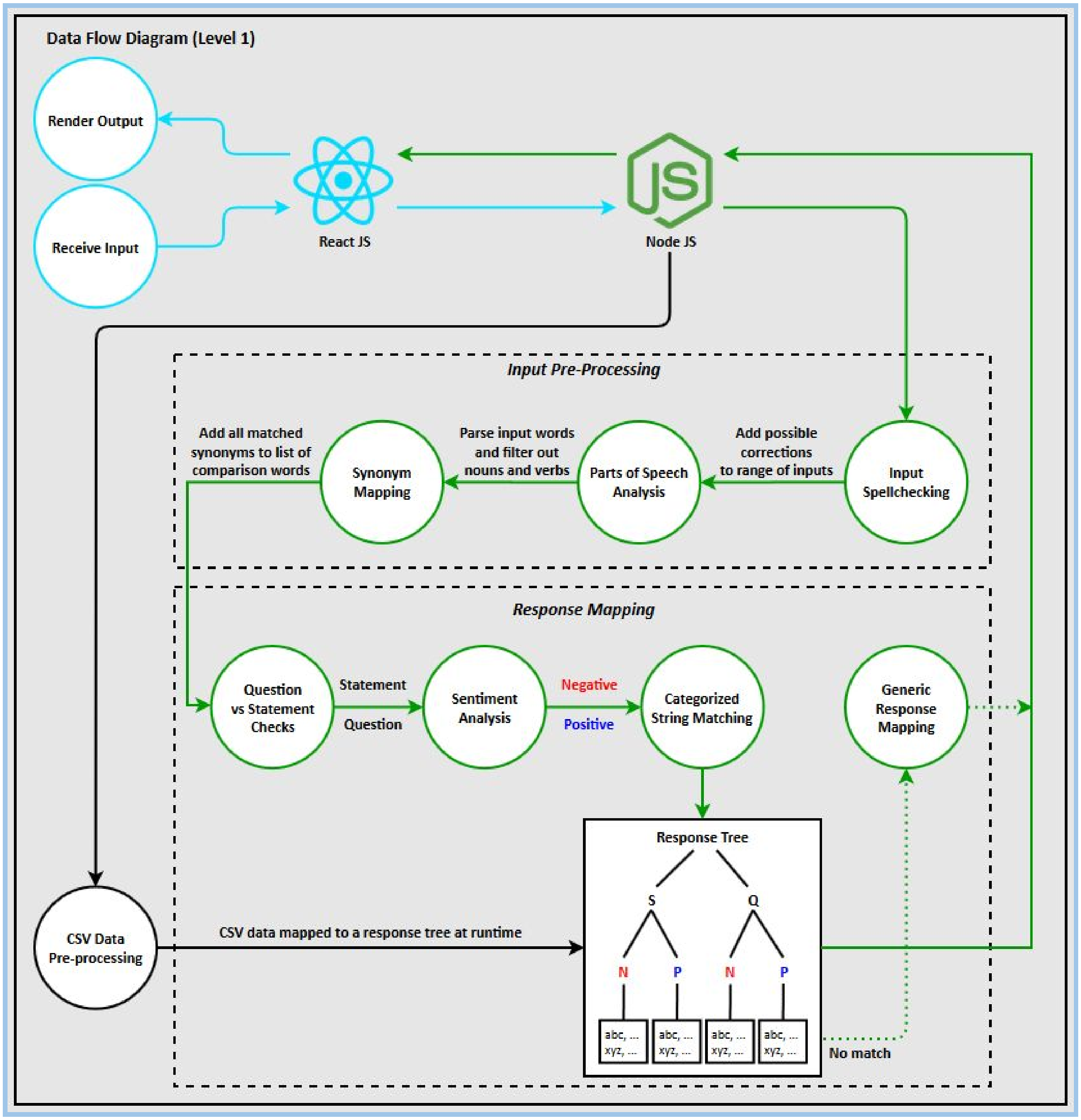
	Name	Predec
1	⊟**A2 - Sprint Backlog 1**	
2	⊟**Requirements**	
3	Research chatbot algorithms	
4	⊟**Code**	
5	Create base application	
6	⊟**Test**	
7	Run base application	5
8	⊟**Documentation**	
9	⊟**Setup environment**	
10	Create WBS / GANTT template	
11	Setup Github repository	
12	Create README template	
13	Write wiki page for react.js	5
14	⊟**A2 - Sprint Backlog 2**	
15	⊟**Requirements**	
16	Setup web server to host production site	
17	Sample conversation 1	
18	Initial text file for output tree	
19	⊟**Design**	
20	Design high-level processing algorithm	
21	⊟**Code**	
22	Implement text pre-processing	
23	Implement text input processing	
24	⊟**Test**	
25	Test web hosting	16
26	⊟**Documentation**	
27	Add SDLC rational to project document	
28	⊟**A2 - Sprint Backlog 3**	
29	⊟**Requirements**	
30	Align requirements with deliverables	
31	⊟**Code**	
32	Text mapping algorithm	
33	Basic text input and output UI	
34	Output generation	32
35	⊟**Test**	
36	Test and prepare for 30 turn conversation	
37	Test and review code changes	34
38	⊟**Documentation**	
39	⊟**Finalize project document**	
40	Add description to document	
41	Add SDLC phases / tasks to document	
42	Add final WBS / GANTT to document	
43	Add list of limitations to document	
44	Populate sample output file in Github	34
45	Add Github commits by user	
46	Finalize README file in Github repository	30



	Name	Predec
47	⊟Project - Sprint Backlog 1	
48	⊟Requirements	
49	Learn natural NLP functionality and use	
50	Research neural net/machine learning	
51	⊟Code	
52	User input NLP pre-processing	
53	Add weighted logic to selection from response list	
54	Save history of past responses	
55	Start advanced GUI	
56	⊟Project - Sprint Backlog 2	
57	⊟Requirements	
58	Learn about React states and components	
59	Learn about slack integration	
60	⊟Code	
61	Update isQuestion method	
62	Add server-side express functionality	
63	Add scroll bar to chat UI	
64	Finalize UI message styling	
65	⊟Project - Sprint Backlog 3	
66	⊟Design	
67	Design advanced response mapping	
68	⊟Code	
69	Implement Slack chat-bot integration	
70	Connect GUI to message history	
71	Add auto-scroll function to GUI	63
72	Implement natural functions server-side	
73	Expand tree node input functionality	72
74	Implement component-based GUI	
75	Integrate GUI with server-side functionality	74
76	⊟Test	
77	Test and troubleshoot new master branch	75
78	Test natural functions server-side	72
79	⊟Project - Sprint Backlog 4	
80	⊟Design	
81	Add extra topic to agent repertoire	
82	Add 5 non-topic related responses	
83	⊟Code	
84	Implement feature to handle spelling mistakes	
85	Add synonyms to language processing	
86	⊟Test	
87	Test pre-processing for bugs	85,84
88	⊟Documentation	
89	Add list of 5 features that can be used as API	
90	Walk through of each feature in README file	
91	Provide level 0 DFD	90
92	Provide level 1 DFD	90
93	⊟Presentation	
94	Begin writing script	
95	⊟Project - Sprint Backlog 5	
96	⊟Code	
97	Increase conversation repertoire / capability	
98	Fix bugs found in pre-processing test	87
99	⊟Test	
100	Test and deploy final version on production server	97
101	⊟Documentation	
102	Update README file in Github	
103	Add Github link and commits by user	
104	Finalize and submit documentation	
105	Add list of limitations to document	
106	⊟Presentation	
107	⊟Create and edit a 60 to 90 second video	
108	Description of the conversational topic	
109	Show each feature and how it adds to system	
110	Description of the data flow diagrams	91,92

Data Flow Diagrams:





Limitations:

- Can get very different/unrelated responses when asking the same question or inputting something that falls under two separate and very different tags in the tree. Does not handle multi-part questions.

user: Do you like princesses? How about unicorns? Where do you live?

bot: In the greatest country on Earth, with the greatest economy in the world, so much stronger than China's, the United States of America.

- Does not work with phrases/more than one word when selecting a response from the response tree. This severely limits the bots understanding about what the user is saying.

user: South Korea is doing a great job.

bot: That Kim Jong Un is an interesting guy. He sure has control over there.

- Defines if input is a statement or a question based on whether or not it ends in a "?"

- Sentiment analyzer cannot handle negations.
  - For example "not pleasant" is a positive phrase.

- Responses are based on a manually written CSV file.
  - At the moment each word is mapped on four lines in the csv with two binary values for question/statement and positive/negative.
  - Adding in more factors would lead to an exponential growth and would make the development and maintenance of the csv unmanageable.

Sample Output:

- A discussion with the US president about politics and campaigns

user: Hi

bot: Hi. My name is Donald Trump. I am the President of the United States.

user: Hi donald

bot: You can call me Mr. President.

user: Ok mr president

bot: That's better.

user: How's your 2020 campaign going?

bot: It's going great. It's always great.

user: Do you think you will still be president after all this coronavirus madness?

bot: I'm the greatest president this country has ever had.

user: It's not going to hurt the economy too much?

bot: We have had the strongest economy during my presidency in all of American history. I am the greatest president. Just great.

user: Ok then. What do you think about Joe Biden?

bot: I think Sleepy Joe is losing it.

user: How about Bernie Sanders?

bot: Crazy Bernie is a socialist.

user: He's not a socialist.

bot: All democrats are socialists.

user: I'm going to vote for a democrat.

bot: Vote for me.

user: You're crazy.

bot: Do you know who's crazy? Crazy Bernie Sanders.

user: How's your family?

bot: Family is important. Almost as important as money.

user: Money isn't that important

bot: Vote Trump 2020! Where do you live? Can you vote for me?

user: I live in Kelowna

bot: Where?

user: in Canada

bot: I love Canada. It's so pretty. I like to look at pictures of it. I wouldn't want to live in an igloo though.

user: We don't live in igloos

bot: Did you see Fox News last night?

user: I don't watch fox news

bot: Don't listen to the Fake News media.

user: What do you have against the media?

bot: That left wing media is out to get me.

user: It's not a conspiracy man

bot: What will the democrats come up with next?

user: They will probably just remind the country we're entering a recession

bot: America is resilient. We will not have a recession.

user: Oh the world is going downhill

bot: This is a great world we live in. I wish everyone saw how great I am.

user: It's hard to see past that hair

bot: Fake news!

user: You don't make sense

bot: Can't you read?

user: I can read quite well. You're an idiot

bot: I'm the most intelligent person in all of the United States. I had Elon Musk advising me and he couldn't even keep up.

user: Elon quit on you because you're a psycopath. This conversation is painful

bot: I'm really enjoying this conversation.

user: Of course you are. Ugh

bot: Ya it hurts.

user: How are you're investments doing?

bot: I'm the greatest investor in the world. I make so much money on my investments. No one in the world is as great an investor as me.

user: so are you rich?

bot: I have so much money. Like an incredible amount of money. Not like this country. We're in so much debt because of Obama.

user: back off obama

bot: Obama is the worst. Was he even born in America?

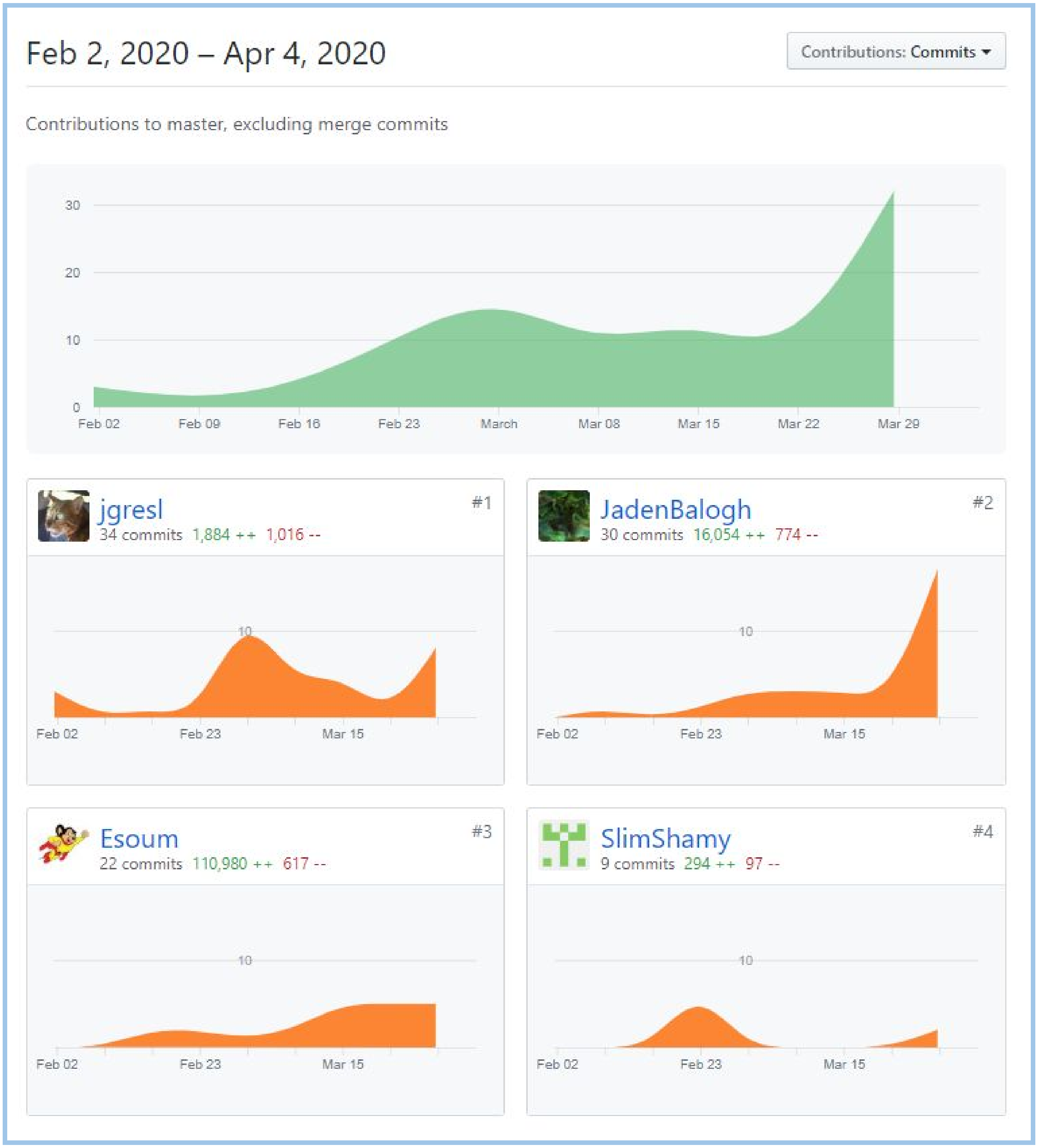
user: wow I wish you were impeached

bot: That's all the time I have. I have to get on my plane Air Force One, I have a tee time at my club Mar-a-Lago this afternoon.

user: ok bye then

bot: Bye-bye

Commits by User:



Feb 2, 2020 – Apr 4, 2020

Contributions to master, excluding merge commits

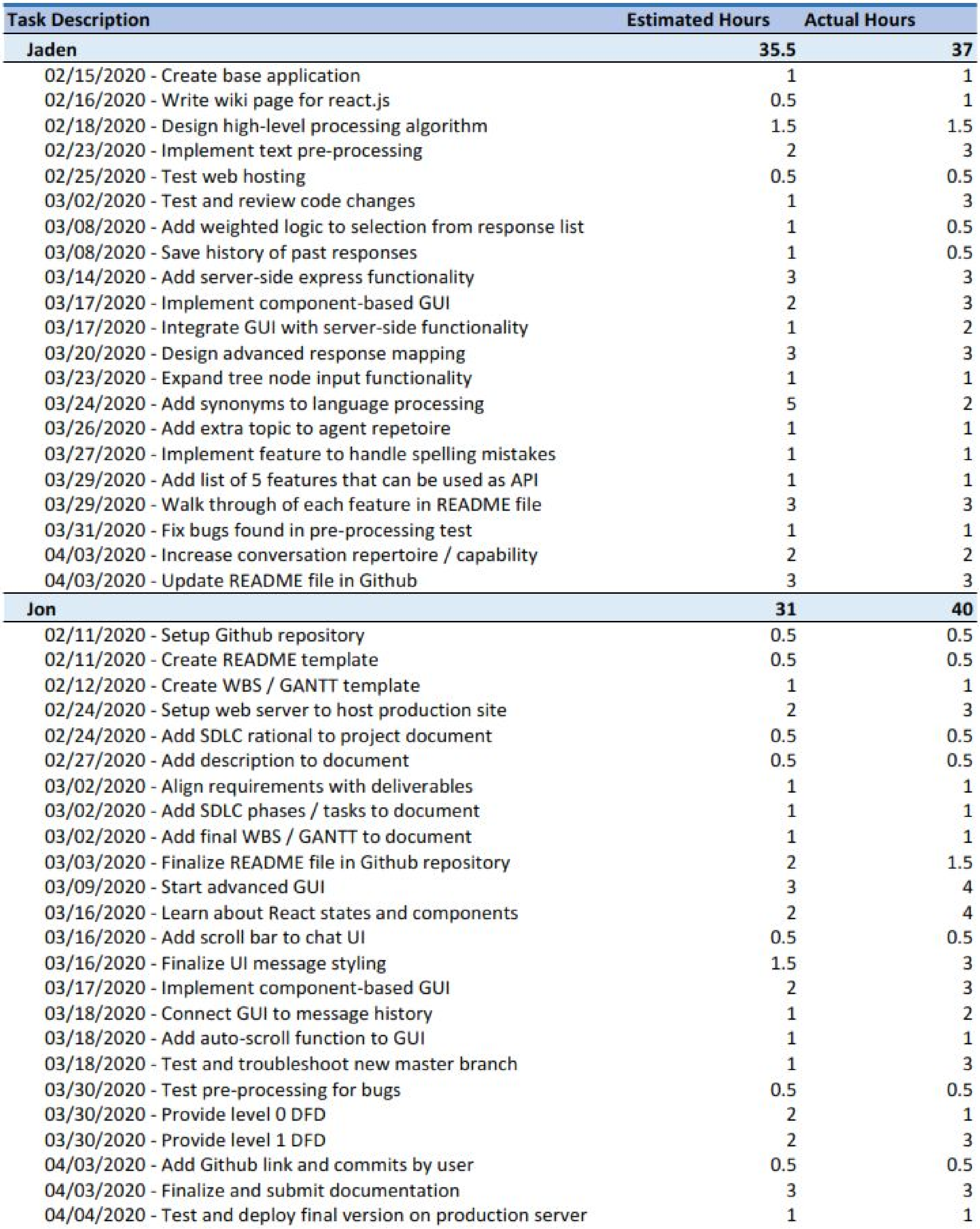
jgresl #1
34 commits 1,884 ++ 1,016 --

JadenBalogh #2
30 commits 16,054 ++ 774 --

Esoum #3
22 commits 110,980 ++ 617 --

SlimShamy #4
9 commits 294 ++ 97 --

Task Hour Summary:



Task Description	Estimated Hours	Actual Hours
Jaden	**35.5**	**37**
02/15/2020 - Create base application	1	1
02/16/2020 - Write wiki page for react.js	0.5	1
02/18/2020 - Design high-level processing algorithm	1.5	1.5
02/23/2020 - Implement text pre-processing	2	3
02/25/2020 - Test web hosting	0.5	0.5
03/02/2020 - Test and review code changes	1	3
03/08/2020 - Add weighted logic to selection from response list	1	0.5
03/08/2020 - Save history of past responses	1	0.5
03/14/2020 - Add server-side express functionality	3	3
03/17/2020 - Implement component-based GUI	2	3
03/17/2020 - Integrate GUI with server-side functionality	1	2
03/20/2020 - Design advanced response mapping	3	3
03/23/2020 - Expand tree node input functionality	1	1
03/24/2020 - Add synonyms to language processing	5	2
03/26/2020 - Add extra topic to agent repetoire	1	1
03/27/2020 - Implement feature to handle spelling mistakes	1	1
03/29/2020 - Add list of 5 features that can be used as API	1	1
03/29/2020 - Walk through of each feature in README file	3	3
03/31/2020 - Fix bugs found in pre-processing test	1	1
04/03/2020 - Increase conversation repertoire / capability	2	2
04/03/2020 - Update README file in Github	3	3
Jon	**31**	**40**
02/11/2020 - Setup Github repository	0.5	0.5
02/11/2020 - Create README template	0.5	0.5
02/12/2020 - Create WBS / GANTT template	1	1
02/24/2020 - Setup web server to host production site	2	3
02/24/2020 - Add SDLC rational to project document	0.5	0.5
02/27/2020 - Add description to document	0.5	0.5
03/02/2020 - Align requirements with deliverables	1	1
03/02/2020 - Add SDLC phases / tasks to document	1	1
03/02/2020 - Add final WBS / GANTT to document	1	1
03/03/2020 - Finalize README file in Github repository	2	1.5
03/09/2020 - Start advanced GUI	3	4
03/16/2020 - Learn about React states and components	2	4
03/16/2020 - Add scroll bar to chat UI	0.5	0.5
03/16/2020 - Finalize UI message styling	1.5	3
03/17/2020 - Implement component-based GUI	2	3
03/18/2020 - Connect GUI to message history	1	2
03/18/2020 - Add auto-scroll function to GUI	1	1
03/18/2020 - Test and troubleshoot new master branch	1	3
03/30/2020 - Test pre-processing for bugs	0.5	0.5
03/30/2020 - Provide level 0 DFD	2	1
03/30/2020 - Provide level 1 DFD	2	3
04/03/2020 - Add Github link and commits by user	0.5	0.5
04/03/2020 - Finalize and submit documentation	3	3
04/04/2020 - Test and deploy final version on production server	1	1

Task Description	Estimated Hours	Actual Hours
Michael	**39**	**36.5**
02/13/2020 - Research chatbot algorithms	4	4
02/17/2020 - Run base application	0.5	0.5
02/18/2020 - Design high-level processing algorithm	1.5	1.5
02/22/2020 - Sample conversation 1	1	1.5
02/24/2020 - Initial text file for output tree	0.5	0.5
02/27/2020 - Add list of limitations to document	0.5	0.5
02/28/2020 - Test and prepare for 30 turn conversation	1	2
02/28/2020 - Populate sample output file in Github	0.5	0.5
03/02/2020 - Add Github commits by user	0.5	0.5
03/04/2020 - Learn natural NLP functionality and use	3	3
03/09/2020 - Research neural net/machine learning	2	1
03/09/2020 - User input NLP pre-processing	2	2
03/19/2020 - Implement natural functions server-side	2	2
03/20/2020 - Design advanced response mapping	3	3
03/21/2020 - Test natural functions server-side	1	1
03/24/2020 - Add synonyms to language processing	5	2
03/26/2020 - Add extra topic to agent repetoire	1	1
03/27/2020 - Add 5 non-topic related responses	0.5	0.5
03/27/2020 - Implement feature to handle spelling mistakes	1	1
03/29/2020 - Walk through of each feature in README file	3	3
04/03/2020 - Increase conversation repertoire / capability	2	2
04/03/2020 - Update README file in Github	3	3
04/03/2020 - Add list of limitations to document	0.5	0.5
Shamus	**32.5**	**32.5**
02/13/2020 - Research chatbot algorithms	4	4
02/18/2020 - Design high-level processing algorithm	1.5	1.5
02/22/2020 - Implement text input processing	3	4
02/26/2020 - Text mapping algorithm	3	3
02/27/2020 - Basic text input and output UI	2	1
02/27/2020 - Output generation	1	1
03/04/2020 - Learn natural NLP functionality and use	3	3
03/10/2020 - Update isQuestion method	1	1
03/15/2020 - Learn about slack integration	4	4
03/19/2020 - Implement Slack chat-bot integration	3	3
03/30/2020 - Begin writing script	1	1
04/02/2020 - Create and edit a 60 to 90 second video	3	3
04/02/2020 - Description of the conversational topic	1	1
04/02/2020 - Show each feature and how it adds to system	1	1
04/02/2020 - Desciption of the data flow diagrams	1	1
Grand Total	**138**	**146**

Evaluation:

Programming Features	**Max**	**Source**
Make a simple GUI so that the user is typing into a nicer interface and can view a recent history of the conversation.	5	http://50.98.99.115:5000/ src/App.css src/App.js src/components/*.js
Add an extra topic to your agent's repertoire. Ensure this topic has similarities with the original topic. For example, if your original topic is volleyball, you may want to add basketball as a second topic.	2	public/treeinput.csv
Add a feature that enables your agent to give at least 5 different reasonable responses when the user enters something outside the two topics.	3	src/server.js public/treeinput.csv
Add a feature that enables your agent to handle spelling mistakes of the words that your agent is supposed to recognize.	5	src/server.js
Use of language toolkits, incorporate feature to improve your conversation: ➔ Synonym recognition - WordNet ➔ POS tagging - Stanford toolkit, OpenNLP ➔ Sentiment analysis - Stanford toolkit	 10 10 10	src/server.js
Other features discussed with Dr. Apurva Nayaran: ➔ Slack bot integration	15	README.md
Documentation	**Max**	**Source**
README file in your repository describing what you've done.	2	README.md
At the end of your README file, include: ➔ a list of each feature you programmed for this assignment ➔ for each item on that list, explain briefly how you used that feature to improve your agent's conversation or your overall system ➔ for each explanation, give a snippet of a conversation that demonstrates your feature	5	README.md documentation/images/*
Provide a Level 0 DFD for your system with description.	3	documentation/DFD.pdf
Provide a Level 1 DFD for your system with description.	5	documentation/DFD.pdf
Submission of your GitHub repository. Graph showing different features developed on a separate branch and the commits made in the repository.	5	https://github.com/jgresl/trump-bot
Include sample output in your project report. Have one dialogue (at least 30 turns) that show a good or feasible conversation -- ensure your new features are demonstrated! Document a list of limitations of your program, and have at least two short dialogues that show when your agent is not able to handle the conversation properly.	5	README.md sampleconvo.txt sampleconvo2.txt testedoutput.txt [page 13]
Based on your system, include a list of at least 5 features that you can extract from your code or design that can be shared with others as an API.	5	[page 3]
Presentation	**Max**	**Source**
A 60 to 90 second video of your assignment showing: ➔ A brief description of your program's conversational topic ➔ Each feature you've programmed and how you used it to improve either the conversation or the overall system (since A2) ➔ A description of your DFDs ➔ You should narrate and/or provide subtitles	5	https://youtu.be/hvAclW_ebs8