ENSE 470, Milestone 7

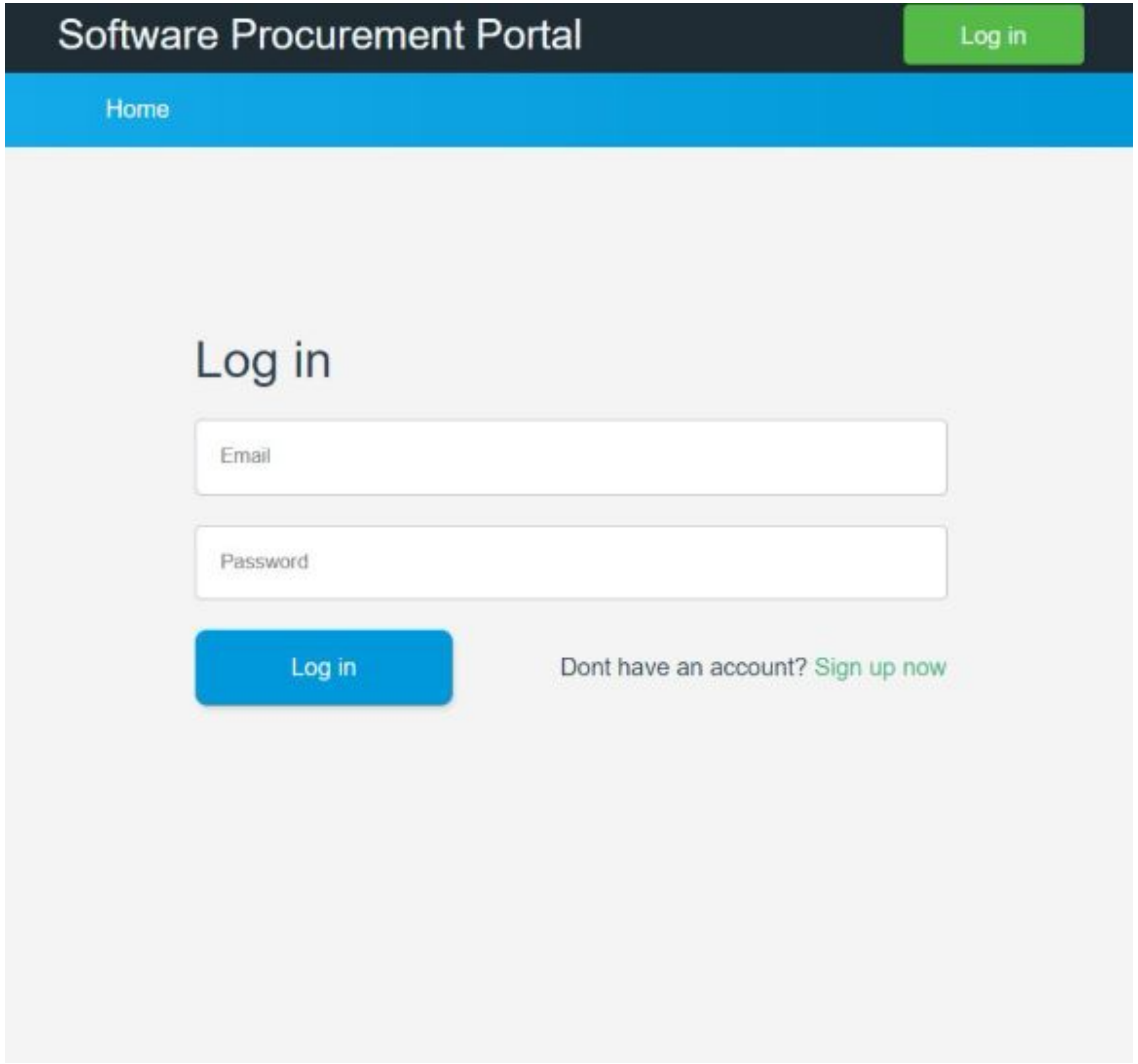
Team DCC

Dakota Fisher, Chengyu Lou, Connor Meredith

April 12th, 2018

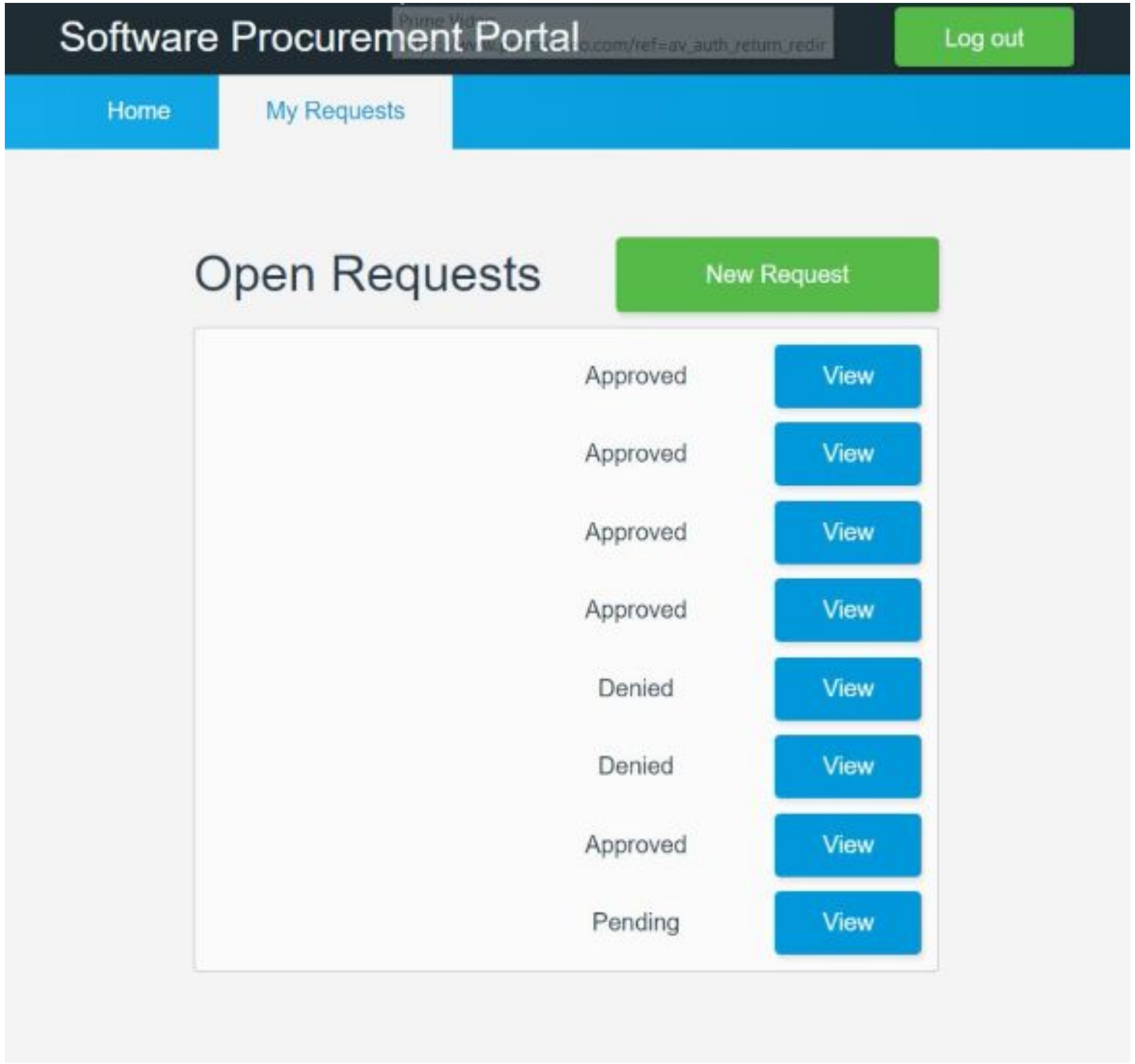
Introduction

- Team under review: ENSE Master Race



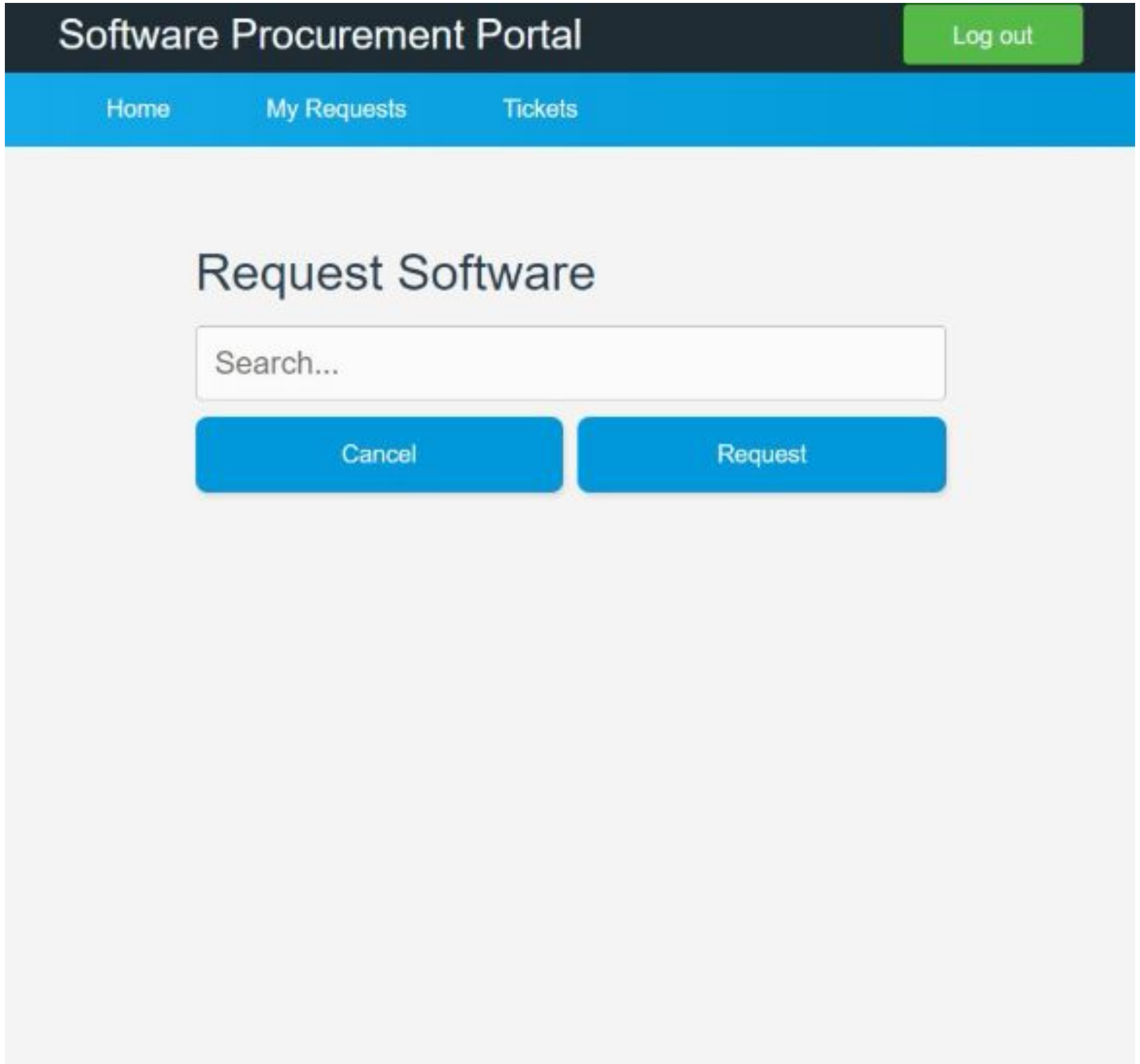
Log in

Dont have an account? Sign up now

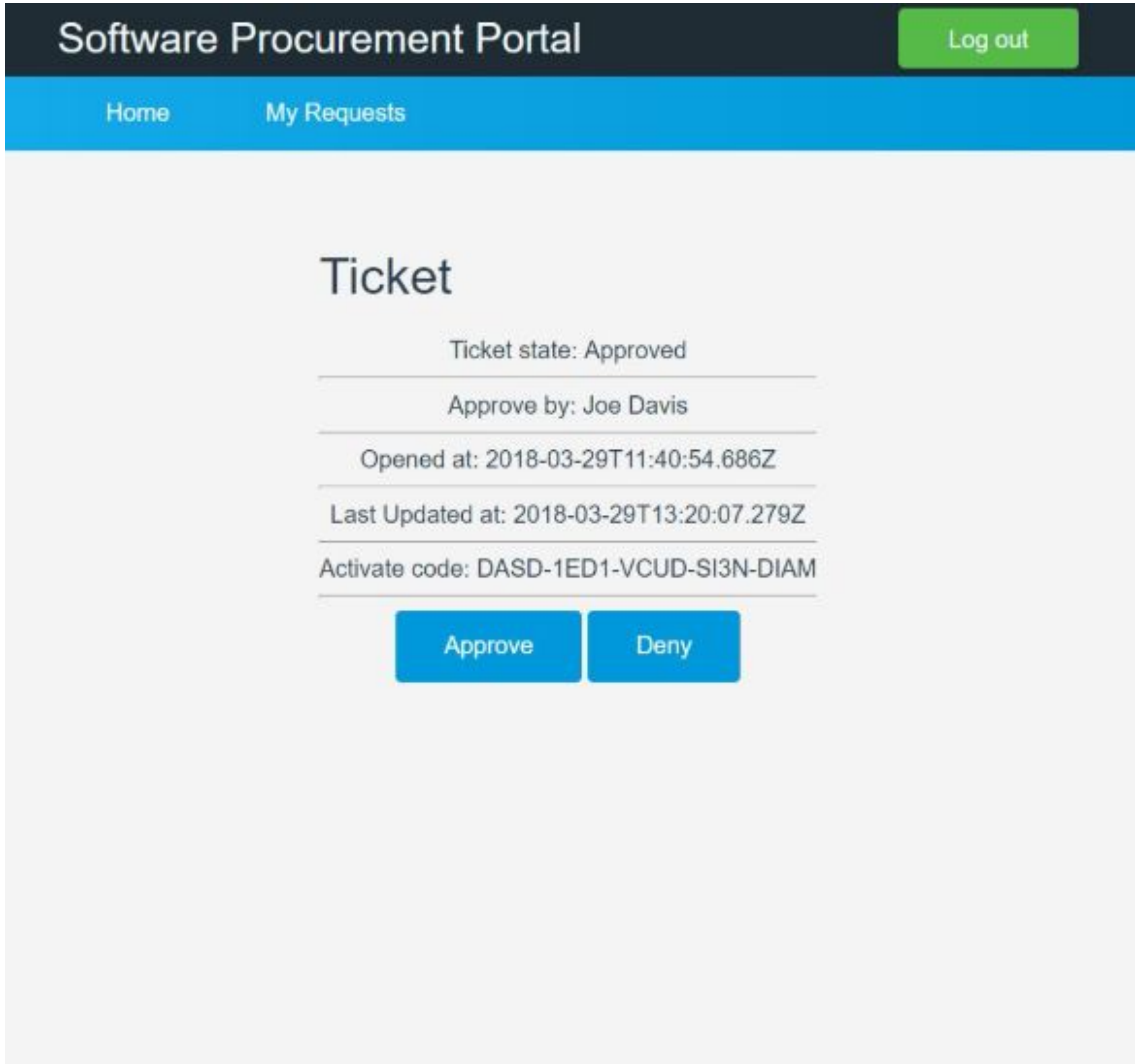


Open Requests

Approved
Approved
Approved
Approved
Denied
Denied
Approved
Pending



Request Software



Ticket

Ticket state: Approved

Approve by: Joe Davis

Opened at: 2018-03-29T11:40:54.686Z

Last Updated at: 2018-03-29T13:20:07.279Z

Activate code: DASD-1ED1-VCUD-SI3N-DIAM

Team GitHub review

- Is the team’s GitHub readable?
  - Yes,the github is readable, albeit programmed in a foreign language.
  - https://github.com/wellsjonathon/ense470
- Does the team provide good documentation?
  - Code is quite readable. Vue kinda documents itself.
  - No real need for in-depth commenting (descriptive expressions)
- Is your team able to understand the file/folder names/structure (flow)?
  - Vue.js has its own file structure to get used to but, it seems to make sense.

Team GitHub review

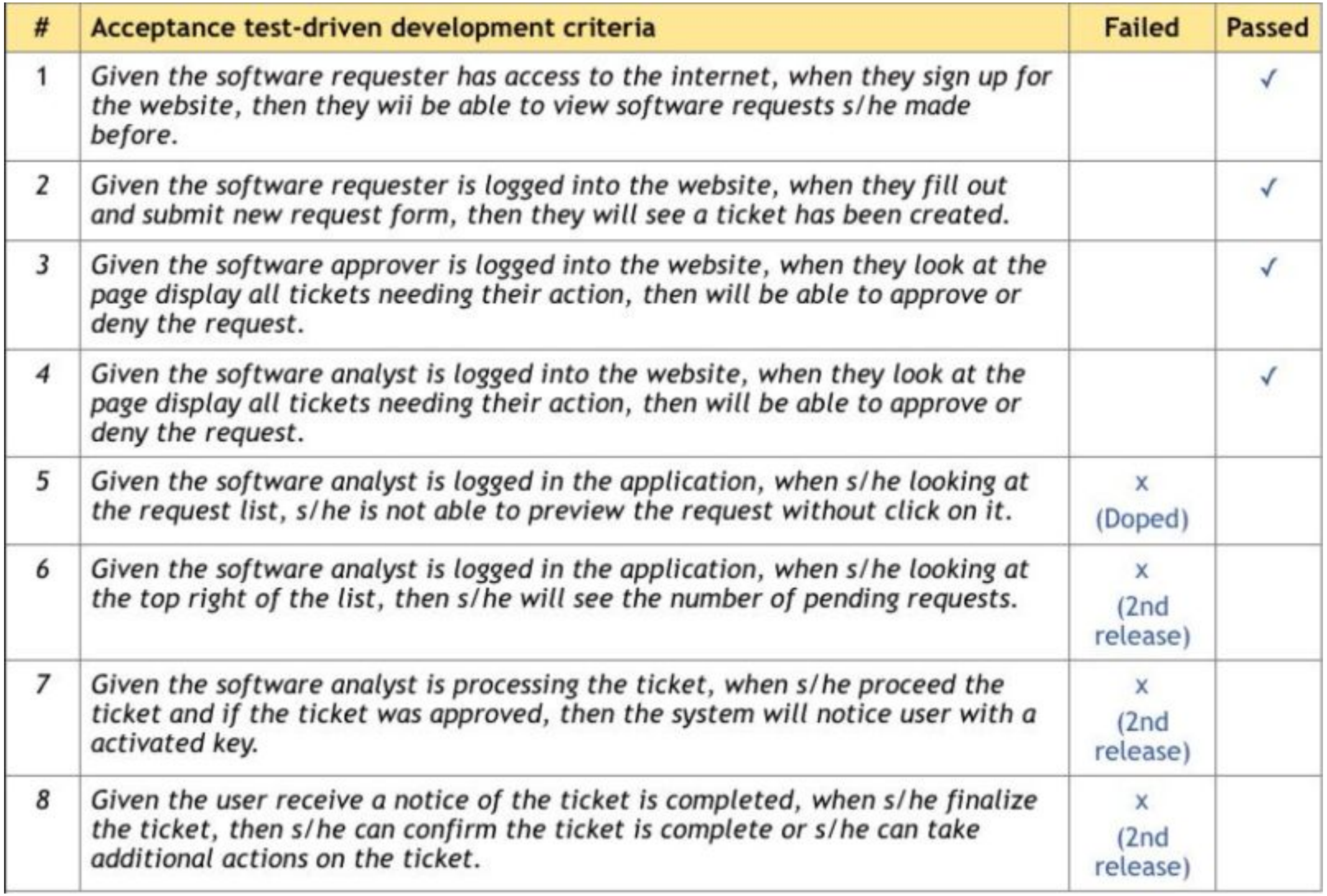
- Would your team be able to easily and quickly pick up where they left off?
  - Assuming we learned or understood Vue.js, we most likely could.
- Do you have any comments/guidance for the team in this area?
  - More readme’s and explanations for folders would be helpful.

Refactor review (High-level focus)

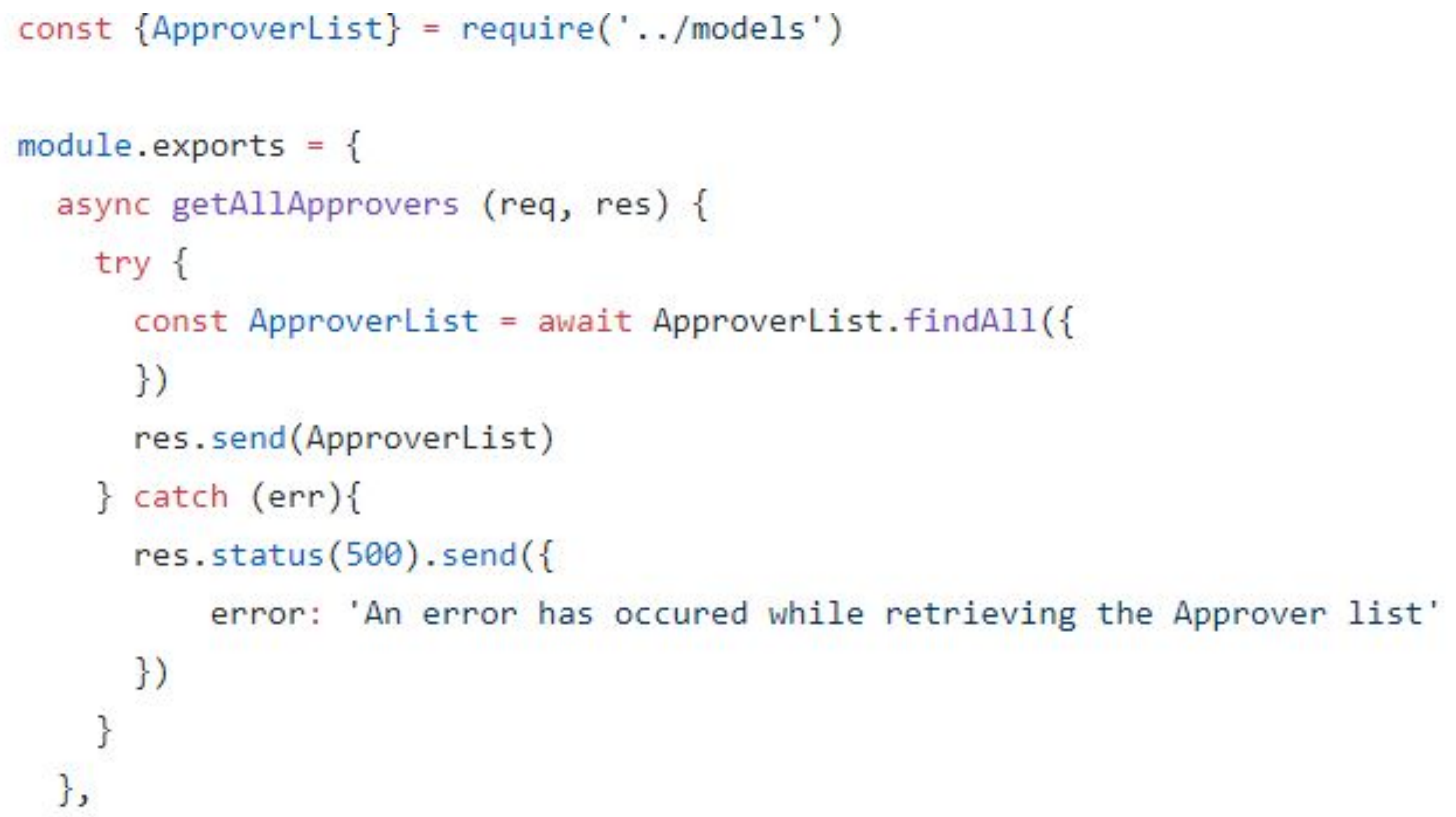
- Is the team’s code obvious/readable for other programmers
  - Will others understand what’s going on? Yes
  - Does your team understand what is going on? Yes
  - Does the code provided as-is appear easy to maintain? – yes/no – Why? Yes
    - Clean, descriptive names, code is well organized and compartmentalized.

Refactor review (High-level focus)

- Do all tests pass?
  - Their ATDD’s were all stored within their milestone report out.
    - Fairly easy to find.
  - The ATDD’s they claimed to have passed, did indeed pass. The others had explicitly been delayed.
  - They seem to have hit all the key ATDD’s for their user story map.

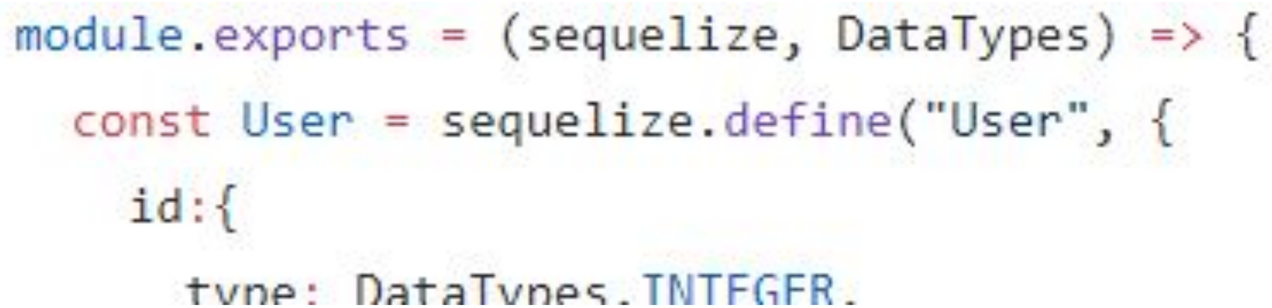


#	Acceptance test-driven development criteria	Failed	Passed
1	*Given the software requester has access to the internet, when they sign up for the website, then they wii be able to view software requests s/he made before.*		✓
2	*Given the software requester is logged into the website, when they fill out and submit new request form, then they will see a ticket has been created.*		✓
3	*Given the software approver is logged into the website, when they look at the page display all tickets needing their action, then will be able to approve or deny the request.*		✓
4	*Given the software analyst is logged into the website, when they look at the page display all tickets needing their action, then will be able to approve or deny the request.*		✓
5	*Given the software analyst is logged in the application, when s/he looking at the request list, s/he is not able to preview the request without click on it.*	x (Doped)	
6	*Given the software analyst is logged in the application, when s/he looking at the top right of the list, then s/he will see the number of pending requests.*	x (2nd release)	
7	*Given the software analyst is processing the ticket, when s/he proceed the ticket and if the ticket was approved, then the system will notice user with a activated key.*	x (2nd release)	
8	*Given the user receive a notice of the ticket is completed, when s/he finalize the ticket, then s/he can confirm the ticket is complete or s/he can take additional actions on the ticket.*	x (2nd release)	



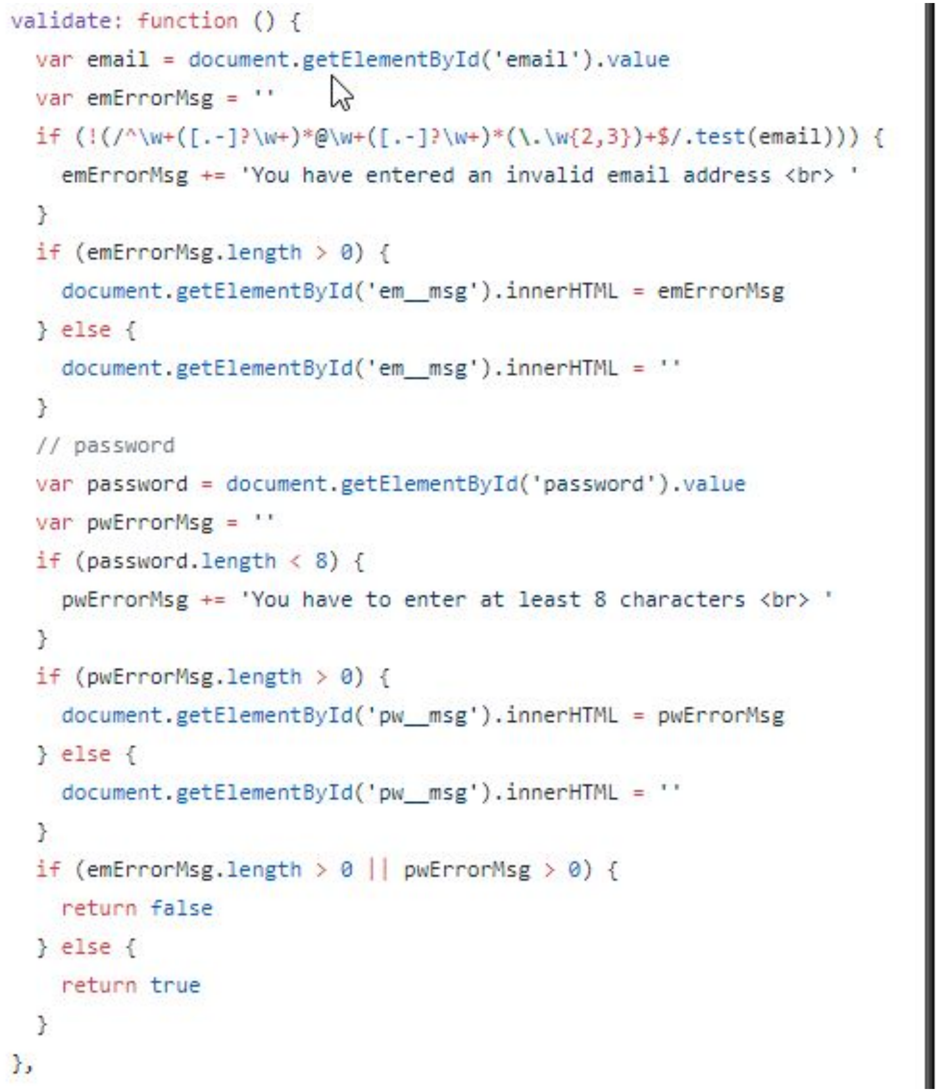
```
const {ApproverList} = require('../models')

module.exports = {
  async getAllApprovers (req, res) {
    try {
      const ApproverList = await ApproverList.findAll({
      })
      res.send(ApproverList)
    } catch (err){
      res.status(500).send({
          error: 'An error has occured while retrieving the Approver list'
      })
    }
  },
```

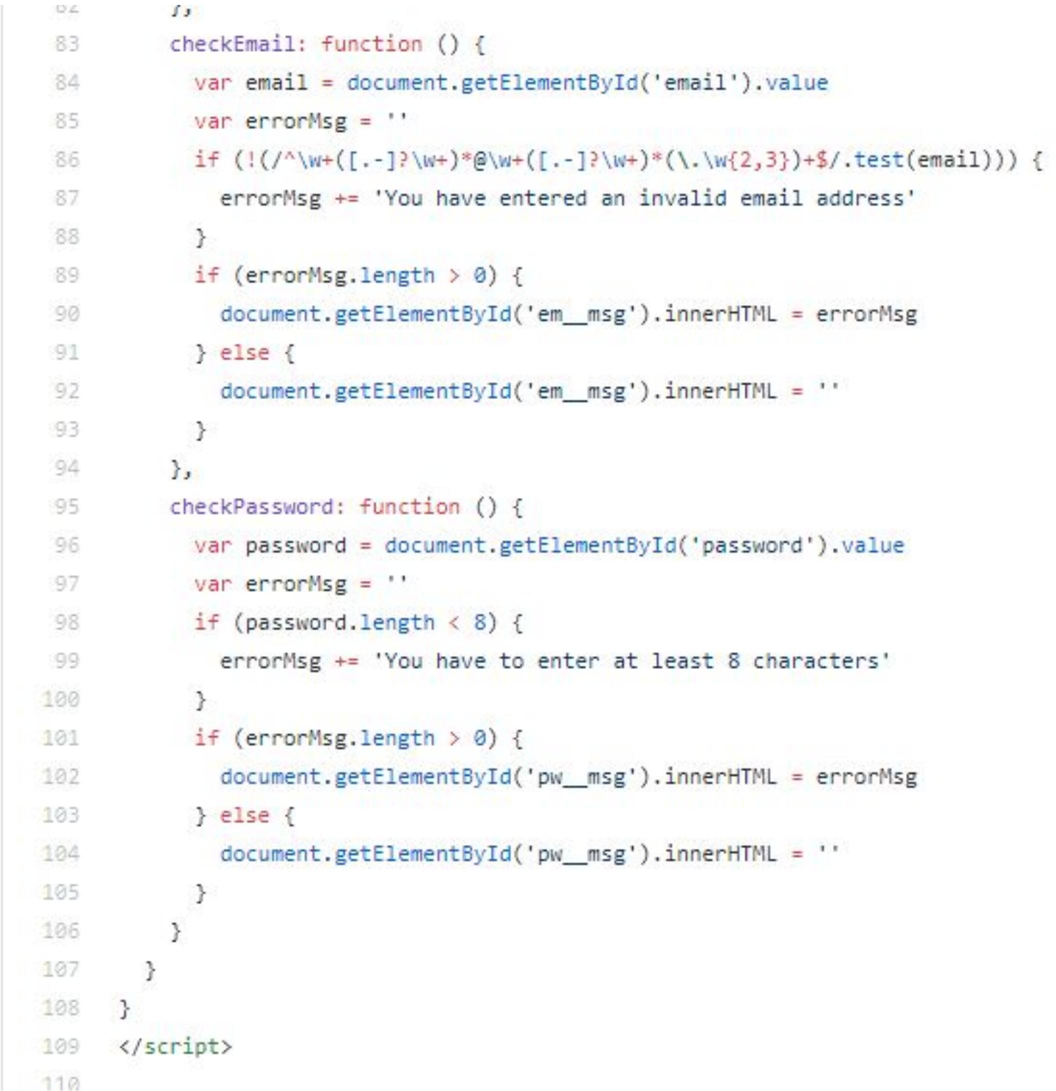


```
module.exports = (sequelize, DataTypes) => {
  const User = sequelize.define("User", {
    id:{
      type: DataTypes.INTEGER,
```

Code Smells (Bloater + Change Preventer)



```
validate: function () {
  var email = document.getElementById('email').value
  var emErrorMsg = ''
  if (!(/^\w+([.-]?\w+)*@\w+([.-]?\w+)*(\.\w{2,3})+$/.test(email))) {
    emErrorMsg += 'You have entered an invalid email address <br> '
  }
  if (emErrorMsg.length > 0) {
    document.getElementById('em__msg').innerHTML = emErrorMsg
  } else {
    document.getElementById('em__msg').innerHTML = ''
  }
  // password
  var password = document.getElementById('password').value
  var pwErrorMsg = ''
  if (password.length < 8) {
    pwErrorMsg += 'You have to enter at least 8 characters <br> '
  }
  if (pwErrorMsg.length > 0) {
    document.getElementById('pw__msg').innerHTML = pwErrorMsg
  } else {
    document.getElementById('pw__msg').innerHTML = ''
  }
  if (emErrorMsg.length > 0 || pwErrorMsg > 0) {
    return false
  } else {
    return true
  }
},
```



```
    },
    checkEmail: function () {
      var email = document.getElementById('email').value
      var errorMsg = ''
      if (!(/^\w+([.-]?\w+)*@\w+([.-]?\w+)*(\.\w{2,3})+$/.test(email))) {
        errorMsg += 'You have entered an invalid email address'
      }
      if (errorMsg.length > 0) {
        document.getElementById('em__msg').innerHTML = errorMsg
      } else {
        document.getElementById('em__msg').innerHTML = ''
      }
    },
    checkPassword: function () {
      var password = document.getElementById('password').value
      var errorMsg = ''
      if (password.length < 8) {
        errorMsg += 'You have to enter at least 8 characters'
      }
      if (errorMsg.length > 0) {
        document.getElementById('pw__msg').innerHTML = errorMsg
      } else {
        document.getElementById('pw__msg').innerHTML = ''
      }
    }
  }
}
</script>
```

Design pattern discussion (High-level focus)

- Design patterns used:
  - State
  - Singleton
  - Decorator
  - Composite
  - Mediator
  - Observer

- Are design patterns used appropriately?
  - Yes
  - Used a good front-end framework for implementing certain design patterns

Design pattern discussion (High-level focus)

- Vue.js seems to have been particularly useful for implementing the state and composite design pattern
  - Built in component system which allows for large-scale applications composed of small, self-contained, and often reusable components
  - Also used built in state management library, vuex, which is very helpful for implementing the state design pattern

Reviewed team’s response

- Ask the reviewed team if they have any questions or comments. This in the spirit of positive non/critical discussion and/or debate

Group reflection

- How did you feel about this milestone? What did you like about it? What did you dislike?
  - We thought it was cool to dive into another’s code, much more doable than the previous course.
- What did you learn about yourself as you collaborated and worked through this milestone?
  - We seemed to pick up on the code layout fairly quickly, considering it was new technology to us.

Group reflection

- How will you use what you have learned going forward?
  - We’ll look for alternative options for programming languages.
  - The importance of code reviews and easy areas for improvements
  - How to develop easier to refactor code.
- What “stuff & things” related to this milestone would you want help with?
  - How much collaboration is okay? Should we be going in blind?
  - Had to ask for usernames and passwords etc.