**CSCI 5530**

**Team 6**

**Griffin Bryant, Marquise Jennett, Dylan Myers, Mansi Pandya, Everrick Wright**

**Software Testing & Quality Assurance**

**Document**

**Version: 1** **Date: (04/14/2020)**

**System Level Requirements Validation**

Software always exists in a broad system context. Although system requirements may have already been established before a software engineer begins his or her work, it is important to determine that system requirements are complete and consistent. If they are not, "push-back" must occur. That is, the software engineering organization must demand clarification from those who have developed system level requirements. The following check will help you to assess system requirements for completeness and consistency. For this checklist, the more questions that elicit a negative response, the higher the risk that the system model will not provide a solid foundation for software engineering work.

* Have the business context and justification for the system been properly developed?
* Has overall feasibility been addressed?
* Have all stakeholders been identified and polled for agreement?
* Have the overall function and behavior of the system been defined?
* Based on existing documentation/information, do you understand the system in the context of each of the views in the system engineering hierarchy?
* Have system processes been adequately (unambiguously) and consistently defined?
* Is system output and input adequately defined?
* Have system-level assumptions, simplifications, limitations, constraints and preferences been explicitly and unambiguously stated?
* Has simulation been done to demonstrate technological feasibility?
* Has a data architecture been identified?
* Has an application (functional) architecture been defined?
* Has the required technology infrastructure for the system been adequately defined?
* *For business applications:* Have ISP and BAA (SEPA, 5/e, p. 253) been performed?
* Has requirements elicitation been performed at the system level?
* Has the scope of the system been bounded?
* Has business and technical feasibility been assessed?
* Have usage scenarios been created at the system level?
* Has a requirements management process been established for the system?
* Has allocation occurred for all system elements?
* Is the allocation for software reasonable and well-defined?
* Have appropriate traceability tables been developed?
* Has a system model (e.g., a Hatley-Pirbhai model) been developed?

**Software Requirements Specification Checklist**

The Software Requirements Specification is the work product that is often produced as a consequence of the analysis activity. The checklist that follows will help you to assist the quality of this document. . For this checklist, the more questions that elicit a negative response, the higher the risk that the specification will not adequately serve its purpose.

* Do stated goals and objectives for software remain consistent with system goals and objectives?
* Have important interfaces to all system elements been described?
* Have all data objects been described? Have all attributes been identified?
* Do major functions remain within scope and have each been adequately described?
* Have functions been refined (elaborated) to an appropriate level of detail?
* Is information flow adequately defined for the problem domain?
* Are diagrams clear; can each stand alone without supplementary text?
* Is the behavior of the software consistent with the information it must process and the functions it must perform?
* Have events and states been identified?
* Are design constraints realistic?
* Have technological risks been fully defined?
* Have alternative software requirements been considered?
* Have validation criteria been stated in detail; are they adequate to describe a successful system?
* Have inconsistencies, omissions or redundancy been identified and corrected?
* Is the customer contact complete?
* Has the user reviewed the Preliminary User's Manual or prototype?
* Be on the lookout for persuasive connectors (e.g., certainly, therefore, clearly, obviously, it follows that), ask why?
* Watch out for vague terms (e.g., some, sometimes, often, usually, ordinarily, most, mostly); ask for clarification.
* When lists are given, but not completed, be sure all items are understood. Keys to look for: "etc., and so forth, and so on, such as."
* Be sure stated ranges don't contain unstated assumptions (e.g., Valid codes range from 10 to 100. Integer? Real? Hex?)
* Beware of vague verbs such as "handled, rejected, processed, skipped, eliminated." There are many ways they can be interpreted.
* Beware of ambiguous pronouns (e.g., The I/O module communicates with the data validation module and its control flag is set. Whose control flag?)
* Look for statements that imply certainty (e.g., always, every, all, none, never), then ask for proof.
* When a term is explicitly defined in one place, try substituting the definition for other occurrences of the term

**Reviewing the Software Design Model**

The design model resides at the core of the software engineering process. It is the place where quality is built into the software (and the place where quality is assessed. For this checklist, the more questions that elicit a negative response, the higher the risk that the analysis model will adequately serve its purpose. . For this checklist, the more questions that elicit a negative response, the higher the risk that the design model will not adequately serve its purpose.  
*General issues:*

* Does the overall design implement all explicit requirements? Has a traceability table been developed?
* Does the overall design achieve all implicit requirements?
* Is the design represented in a form that is easily understood by outsiders?
* Is design notation standardized? Consistent?
* Does the overall design provide sufficient information for test case design?
* Is the design created using recognizable architectural and procedural patterns?
* Does the design strive to incorporate reusable components?
* Is the design modular?
* Has the design defined both procedural and data abstractions that can be reused?
* Has the design been defined and represented in a stepwise fashion?
* Has the resultant software architecture been partitioned for ease of implementation? Maintenance?
* Have the concepts of information hiding and functional independence been followed throughout the design?
* Has a *Design Specification* been developed for the software?

***For data design:***

* Have data objects defined in the analysis model been properly translated into required data structures?
* Do the data structures contain all attributes defined in the analysis model?
* Have any new data structures and/or attributes been defined at design time?
* How do any new data structures and/or attributes related to the analysis model and to overall user requirements?
* Have the simplest data structures required to do the job been chosen?
* Can the data structures be implemented directly in the programming language of choice?
* How are data communicated between software components? It is communicated by having MVVM(Model -View -View Model) design pattern that takes the UI and organizes it in a way that allows the data to be communicated effectively.
* Do explicit data components (e.g., a database) exist? If so, what is their role? Yes, there is a database that holds information for each user and admin , along with messaging and tutoring.

*For architectural design:*

* Has a library of architectural styles been considered prior to the definition of the resultant software architecture?
* Has architectural tradeoff analysis been performed?
* Is the resultant software architecture a recognizable architectural style?
* Has the architecture been exercised against existing usage scenarios?
* Has an appropriate mapping been used to translate the analysis model into the architectural model?
* Can quality characteristics associated with the resultant architecture (e.g., a factored call-and-return architecture) be readily identified from information provided in the design model?

*For user interface design:*

* Have the results of task analysis been documented?
* Have goals for each user task been identified?
* Has an action sequence been defined for each user task?
* Have various states of the interface been documented?
* Have objects and actions that appear within the context of the interface been defined?
* Have the three "golden rules" (SEPA, 5/e, p. 402) been maintained throughout the GUI design?
* Has flexible interaction been defined as a design criterion throughout the interface?
* Have expert and novice modes of interaction been defined?
* Have technical internals been hidden from the casual user?
* Is the on-screen metaphor (if any) consistent with the overall applications?
* Are icons clear and understandable?
* Is interaction intuitive?
* Is system response time consistent across all tasks?
* Has an integrated help facility been implemented?
* Are all error messages displayed by the interface easy to understand? Do they help the user resolve the problem quickly?
* Is color being used effectively?
* Has a prototype for the interface been developed?
* Have user's impressions of the prototype been collected in an organized manner?

*For component-level design:*

* Have proof of correctness techniques (SEPA, 5/e, Chapter 26) been applied to all algorithms?
* Has each algorithm been "desk-tested" to uncover errors? Is each algorithm correct?
* Is the design of the algorithm consistent with the data structured that the component manipulates?
* Have algorithmic design alternatives been considered? If yes, why was this design chosen?
* Has the complexity of each algorithm been computed?
* Have structured programming constructs been used throughout?

## Traceability Table:

|  |  |  |
| --- | --- | --- |
| Requirement | Test Case ID | Status |
| The system shall have the ability to log onto the application | TC\_1  TC\_2  TC\_26  TC\_27 | TC\_1 – Pass  TC\_2 – Pass  TC\_26 – Pass  TC\_27 – Pass |
| The system database shall store titles of board members | TC\_29  TC\_28 | TC\_29 – Pass  TC\_28 – Pass |
| The system shall set eboard members as club administrators | TC\_16  TC\_18  TC\_19  TC\_24  TC\_28 | TC\_16 – Pass  TC\_18 – Pass  TC\_19 – Pass  TC\_24 – Pass  TC\_28 – Pass |
| The system shall allow users to view events | TC\_5  TC\_25 | TC\_5 – Pass  TC\_25 – Pass |
| The system shall allow users to open event locations in their default map application | TC\_14 | TC\_14 – Fail |
| The system shall allow users to request a mentor | TC\_3 | TC\_3 – Pass |
| The system shall allow the tutor to be matched by things like class experience | TC\_4 | TC\_4 – Fail |
| The application shall be able to send notifications for meetings and events via email or push notification | TC\_13 | TC\_13 – Fail |
| The system shall prevent messages containing profanity from being sent to in the chat | TC\_15 | TC\_15 – Fail |
| Users shall be able to message other users directly |  |  |
| Users shall be able to send messages inside of groups with set topics |  |  |
| Non-members shall be able see events and create an account | TC\_6  TC\_17 | TC\_6 – Pass  TC\_17 – Pass |
| Users shall have the ability to create a profile | TC\_16  TC\_20  TC\_21  TC\_22  TC\_23 | TC\_16 – Pass  TC\_20 – Pass  TC\_21 – Pass  TC\_22 – Pass  TC\_23 – Pass |
| The system should allow users to see a list of known ACM GSU alumni | TC\_7 TC\_9  TC\_10 | TC\_7 – Pass  TC\_9 – Pass  TC\_10 – Fail |
| The system shall have the ability to keep a tally of the attendance at each meeting | TC\_12 | TC\_12 – Pass |
| The system shall allow a user to be removed from the ACM app after deferring to ACM constitution | TC\_11 | TC\_11 – Fail |
| The system shall allow administrators to create anonymous voting polls for the executive board | TC\_8 | TC\_8 – Pass |

# 

# Testing:

Conducting Unit Tests:

## Requirement Tests:

### Test Case 1:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #:** TC\_1 | | | **Test Case Name:** Sign Up Page | | |
| **System:** Login Page | | | **Subsystem:** Sign Up Page | | |
| **Designed By:** Everrick Wright | | | **Designed Date:** April 19, 2020 | | |
| **Executed By:** Everrick Wright | | | **Execution Date:** April 19, 2020 | | |
| **Short Description:** The system shall allow the mentor to be matches by things like class experience | | | | | |
| **Stp#** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| **1** | User opens ACM application | Dashboard screen is presented | | Pass |  |
| **2** | User Clicks Login Button | Login Screen is presented with opens to sign up or log in | | Pass |  |
| **3** | User enters first & last name, email address, and phone number, password and confirmed password | User receives message that account is created | | Pass | **User account should be created in database** |
| **4** | User is taken to the home page | ACM dashboard | | Pass |  |
| **Post-conditions:**  The user has usability to all normal app features excluding control panel. | | | | | |

### Test Case 2:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_2 | | | Test Case Name: Login | | |
| System: Login Page | | | Subsystem: Sign Up Page | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: User logs into their account and is unable to login | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | User transferred to page with data fields that they are to fill in | | Pass |  |
| 2 | User enters name, email address, and phone number | Application prompts ‘name, email id doesn’t exist’ | | Pass |  |
| 3 | The page directs to the Signup page | User transferred to page with data fields that they are to fill in | | Pass |  |
| 4 | User clicks Submit button | ACM Dashboard | | Pass |  |
| Post-conditions:  The user has ability to all App features | | | | | |

### Test Case 3:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #:** TC\_3 | | | **Test Case Name:** Requesting a mentor | | |
| **System:** Tutoring Requests | | | **Subsystem:** Tutoring Requests | | |
| **Designed By:** Everrick Wright | | | **Designed Date:** April 19, 2020 | | |
| **Executed By:** Everrick Wright | | | **Execution Date:** April 19, 2020 | | |
| **Short Description:** The system shall allow the mentor to be matches by things like class experience | | | | | |
| **Stp#** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| **1** | User Logins | Successful login | | Pass |  |
| **2** | User Clicks “Tutoring” tab | Tutoring screen is presented | | Pass |  |
| **3** | User enters class they need help with & a prospective date | Event should be sent to Tutoring requests screen able to be selected | | Pass | **Data should be stored in database** |
| **4** | User notified of match | Notification sent of match | | Fail | **No system implementation** |
| **Post-conditions:**  N/A | | | | | |

### Test Case 4:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #:** TC\_4 | | | **Test Case Name:** Match by Class Experience | | |
| **System:** Tutoring Requests | | | **Subsystem:** Tutoring Requests | | |
| **Designed By:** Everrick Wright | | | **Designed Date:** April 19, 2020 | | |
| **Executed By:** Everrick Wright | | | **Execution Date:** April 19, 2020 | | |
| **Short Description:** The system shall allow the mentor to be matches by things like class experience | | | | | |
| **Stp#** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| **1** | User Logins | Successful login  Taken to dashboard page | | Pass |  |
| **2** | User Clicks “Tutoring” tab | Tutoring screen is presented | | Pass |  |
| **3** | User enters class they need help with & a prospective date | Event should be sent to Tutoring requests screen able to be selected | | Pass | **Data should be stored in database** |
| **4** | Matching of mentor by class experience | Specific events show if class is marked as taken | | Fail | **No system implementation** |
| **Post-conditions:**  N/A | | | | | |

### 

### Test Case 5:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #:** TC\_5 | | | **Test Case Name:** Events | | |
| **System:** Events | | | **Subsystem:** N/A | | |
| **Designed By:** Everrick Wright | | | **Designed Date:** April 19, 2020 | | |
| **Executed By:** Everrick Wright | | | **Execution Date:** April 19, 2020 | | |
| **Short Description:** The systems shall allow users to view events | | | | | |
| **Stp#** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| **1** | User logins | Successful Login Attempt | | Pass |  |
| **2** | User Clicks Events tab | Events screen is promptly shown with current events | | Pass | **Events pulled from database** |
| **Post-conditions:**  N/A | | | | | |

### 

### Test Case 6:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #:** TC\_6 | | | **Test Case Name:** Non-members view | | |
| **System:** Dashboard | | | **Subsystem:** Events & About Us | | |
| **Designed By:** Everrick Wright | | | **Designed Date:** April 19, 2020 | | |
| **Executed By:** Everrick Wright | | | **Execution Date:** April 19, 2020 | | |
| **Short Description:** Non-members (not logged in) shall be able to see events, about us, and able to create account | | | | | |
| **Stp#** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| **1** | User Opens Application | Dashboard screen is opened, only viewable tabs are events & about us | | Pass |  |
| **2** | User clicks events | Events page opens | | Pass |  |
| **3** | User clicks about us | About us page opens | | Pass |  |
| **4** | User can log in **(see Test Case 1)** | **Test Case 1** | | N/A |  |
| **Post-conditions:**  See test case 1 for step #4 | | | | | |

### Test Case 7:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_7 | | | Test Case Name: Alumni page | | |
| System: Alumni | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: The system shall allow users to see list of known ACM GSU alumni | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User logins | Successful Login Attempt | | Pass |  |
| 2 | User Clicks Alumni tab | List of ACM Alumni is showed | | Pass | **Alumni are pulled from database** |
| Post-conditions:  N/A | | | | | |

### Test Case 8:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_8 | | | Test Case Name: Create polls | | |
| System: Control Panel | | | Subsystem: Polls | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: The system shall allow administrators to create anonymous polls for executive board | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | Admin logins | Successful Login Attempt | | Pass |  |
| 2 | Admin clicks Polls tab, clicks on Create Poll, Options and Post Poll | Poll is posted | | Pass | **Poll is posted on Events page** |
| Post-conditions:  N/A | | | | | |

### Test Case 9:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_9 | | | Test Case Name: Employers page | | |
| System: Alumni | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: The system shall allow users to see list of known ACM GSU alumni | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User logins | Successful Login Attempt  Transferred to home page | | Pass |  |
| 2 | User Clicks Employers tab | List of GSU ACM employers is shown | | Pass | **Alumni are pulled from database** |
| Post-conditions:  N/A | | | | | |

### Test Case 10:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_10 | | | Test Case Name: Employers page | | |
| System: Alumni | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 20, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 20, 2020 | | |
| Short Description: The system should allow users to get in contact with possible employers for internships, co-ops, and jobs | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
|  |  |  | |  |  |
| Post-conditions: | | | | | |

### Test Case 11:

### 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_11 | | | Test Case Name: Remove members | | |
| System: Control Panel | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 20, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow a user to be removed from the ACM app after deferring to ACM constitution | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
|  |  |  | |  |  |
| Post-conditions: | | | | | |

### Test Case 12:

### 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_12 | | | Test Case Name: Attendance | | |
| System: Control Panel | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 20, 2020 | | |
| Executed By: Everrick Wright | | | Execution Date: April 24, 2020 | | |
| Short Description: The system shall have the ability to keep a tally of the attendance at each meeting | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | Admin Successful login | Login attempt pass | | pass |  |
| 2 | Click on “control pane’” tab | Checks to see if admin, if so, control panel page opens | | pass |  |
| 3 | Clicks on attendance button | Taken to attendance page | | pass |  |
| 4 | Enters names of people attending | Names saved and accounted for | | pass |  |
| Post-conditions: | | | | | |

### Test Case 13:

### 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_13 | | | Test Case Name: Notifications | | |
| System: Control Panel | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 20, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 20, 2020 | | |
| Short Description: The application shall be able to send notifications for meetings and events via email or push notification | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
|  |  |  | |  |  |
| Post-conditions: | | | | | |

### Test Case 14:

### 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_14 | | | Test Case Name:Event locations | | |
| System: Events | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 20, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow users to open event locations in their default map application | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
|  |  |  | |  |  |
| Post-conditions: | | | | | |

### Test Case 15:

### 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_15 | | | Test Case Name: Remove profanity messages | | |
| System: Chat | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 20, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 20, 2020 | | |
| Short Description:The system shall prevent messages containing profanity from being sent to in the chat | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
|  |  |  | |  |  |
| Post-conditions: | | | | | |

System Testing:

## **Testing GUIs:**

|  |  |  |
| --- | --- | --- |
| Test Case | Part Tested | Pass/Fail |
| TC 1 | Navigation Bar | Pass |
| TC 2 | Dashboard Button | Pass |
| TC 3 | Dashboard Event Button | Fail |
| TC 4 | Dashboard Chat Button | Fail |
| TC 5 | Dashboard Profile Button | Fail |
| TC 6 | Login Button | Pass |
| TC 7 | Logout Button | Pass |
| TC 8 | Control Panel Button | Pass |
| TC 9 | Events Button | Pass |
| TC 10 | Profile Button | Pass |
| TC 11 | Members Button | Pass |
| TC 12 | Chat Button | Pass |
| TC 13 | Tutoring Button | Pass |
| TC 14 | Polls Button | Pass |
| TC 15 | Alumni Button | Pass |
| TC 16 | Pay Dues Button | Pass |
| TC 17 | About Us Button | Pass |
| TC 18 | Create Events Button | Pass |
| TC 19 | Create Polls Button | Pass |
| TC 20 | Attendance Button | Pass |
| TC 21 | User Titles Button | Pass |
| TC 22 | Update News Button | Pass |
| TC 23 | Create an Event: Event Title | Pass |
| TC 24 | Create an Event: Location | Pass |
| TC 25 | Create an Event: Date | Pass |
| TC 26 | Create an Event: Details | Pass |
| TC 27 | Create an Event: Create Event Button | Pass |
| TC 28 | Create an Event: Cancel Button | Pass |
| TC 29 | Create Poll: Poll Title | Pass |
| TC 30 | Create a Poll: Option Field | Pass |
| TC 31 | Create a Poll: Add Option Button | Pass |
| TC 32 | Create a Poll: X Button | Pass |
| TC 33 | Create a Poll: Post Poll Button | Pass |
| TC 34 | Create a Poll: Cancel Button | Pass |
| TC 35 | View Attendance: Take Attendance Button | Pass |
| TC 36 | View Attendance: Date Field | Pass |
| TC 37 | View Attendance: Cancel Button | Pass |
| TC 38 | View Attendance: Search Button | Pass |
| TC 39 | Take Attendance: Name Field | Pass |
| TC 40 | Take Attendance: Save Button | Pass |
| TC 41 | Take Attendance: Cancel Button | Pass |
| TC 42 | User Titles: First Name Field | Pass |
| TC 43 | User Titles: Last Name Field | Pass |
| TC 44 | User Titles: Cancel Button | Pass |
| TC 45 | User Titles: Search Button | Pass |
| TC 46 | User Titles: Edit User Button | Pass |
| TC 47 | Edit User: Title Field | Pass |
| TC 48 | Edit User: Admin Box | Pass |
| TC 49 | Edit User: Save Button | Pass |
| TC 50 | Create News Item: Title Field | Pass |
| TC 51 | Create News Item: Details Field | Pass |
| TC 52 | Create News Item: Post Button | Pass |
| TC 53 | Create News Item: Cancel Button | Pass |
| TC 54 | Profile: Edit Profile Button | Pass |
| TC 55 | Profile: Change Password Button | Pass |
| TC 56 | Edit Profile: Upload Image Button | Fail |
| TC 57 | Edit Profile: First Name Field | Pass |
| TC 58 | Edit Profile: Last Name Field | Pass |
| TC 59 | Edit Profile: Email Field | Pass |
| TC 60 | Edit Profile: Phone Field | Pass |
| TC 61 | Edit Profile: Cancel Button | Pass |
| TC 62 | Edit Profile: Update Profile Button | Pass |
| TC 63 | Edit Profile: Change Password Button | Fail |
| TC 64 | Change Password: Upload Image Button | Fail |
| TC 65 | Change Password: Current Password Field | Pass |
| TC 67 | Change Password: New Password Field | Pass |
| TC 68 | Change Password: Confirm Field | Pass |
| TC 69 | Change Password: Cancel Button | Pass |
| TC 70 | Change Password: Update Password Button | Pass |
| TC 71 | Members: Members Tab | Pass |
| TC 72 | Members: Mentors Tab | Pass |
| TC 73 | Members: Profile Button | Fail |
| TC 74 | Members: Search Button | Fail |
| TC 75 | Chat: DMs Tab | Pass |
| TC 76 | Chat: Public Channels | Pass |
| TC 77 | Tutoring: Course Title Field | Pass |
| TC 78 | Tutoring: Add Session | Pass |
| TC 79 | Polls: Active Polls Tab | Pass |
| TC 80 | Polls: Past Polls Tab | Pass |
| TC 81 | Polls: Option Buttons | Pass |
| TC 82 | Alumni: Search Button | Fail |
| TC 83 | Alumni: Profile Button | Fail |
| TC 84 | Pay Dues: Cash App Button | Pass |
| TC 85 | Pay Dues: Amazon Pay Button | Pass |
| TC 86 | Pay Dues: PayPal Button | Pass |
| TC 97 | About Us: Learn More Button | Pass |

## Conducting Validation Tests:

### 

### 

### 

### Test Case 16:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_16 | | | Test Case Name: Members added to database | | |
| System: Login | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: User logs into the account | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | User transferred to page with data fields that they are to fill in | | Pass |  |
| 2 | User enters name, email address, and phone number |  | | Pass |  |
| 3 | User clicks Login button | Login success | | Pass |  |
| 4 | Users information is stored into the database | The contents are displayed. Members have access to these pages: Dashboard, events, members, chat tutoring, polls, alumni, pay dues, about us. | | Pass |  |
| Post-conditions:  Users password, name, phone number, and email address are added and saved into the database | | | | | |

### Test Case 17:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_17 | | | Test Case Name: Non-member menu | | |
| System: Main Page | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: Non-member user opens app | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User open apps | App opens to the Main page of the app | | Pass |  |
| 2 | User clicks on the Navigation bar | Because the user is not a member they only have access to the main page, event page, and about page | | Pass |  |
| Post-conditions: | | | | | |

### Test Case 18:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_18 | | | Test Case Name: Member Menu | | |
| System: Login Page | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: Members login | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | User transferred to page with data fields that they are to fill in | | Pass |  |
| 2 | User enters name, email address, and phone number |  | | Pass |  |
| 3 | User clicks Login button | Login success | | Pass |  |
| 4 | User clicks on the Navigation bar | Members have access to these only these pages: Dashboard, events, members, chat tutoring, polls, alumni, pay dues, about us. | | Pass |  |
| Post-conditions:  N/A | | | | | |

### Test Case 19:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_19 | | | Test Case Name: Admin Menu | | |
| System: Control Panel | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow admins to login and view ACM member details | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | Login success | | Pass |  |
| 2 | User clicks on the Navigation tab | Admins have access to all of the member pages and the added page of control panel | | Pass |  |
| Post-conditions:  N/A | | | | | |

### Test Case 20:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_20 | | | Test Case Name: Changing Name | | |
| System: Profile | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow to edit name | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | Login success | | Pass |  |
| 2 | User clicks Profile | Transferred to profile page | | Pass |  |
| 3 | User clicks Edit Profile | Transferred to profile edit page | | Pass |  |
| 4 | User changes name |  | | Pass | **The user's name is changed in the database.** |
| 5 | User clicks Save | User taken back to profile page | | Pass |  |
| Post-conditions:  N/A | | | | | |

### Test Case 21:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_21 | | | Test Case Name: Changing password | | |
| System: Profile | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow user to edit Name | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | Login success | | Pass |  |
| 5 | User clicks Profile | Transferred to profile page | | Pass |  |
| 6 | User clicks Update Password | Transferred to update password page | | Pass |  |
| 7 | User enters old password and new password |  | | Pass | **If current password is incorrect password won't change** |
| 8 | User clicks Update password | User taken back to profile page | | Pass | **Users password is changed in the database** |
| Post-conditions:  N/A | | | | | |

### Test Case 22:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_22 | | | Test Case Name: Changing email | | |
| System: Profile | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date:April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow user to edit Email | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | Login success | | Pass |  |
| 2 | User clicks profile | Transferred to profile page | | Pass |  |
| 3 | User clicks edit profile | Transferred to profile edit page | | Pass |  |
| 4 | User changes Email |  | | Pass |  |
| 5 | User clicks Save | User taken back to profile page | | Pass | **Users email is changed in the database** |
| Post-conditions:  N/A | | | | | |

### Test Case 23:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_23 | | | Test Case Name: Changing phone number | | |
| System: Profile | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow user to edit Phone Number | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | Login success | | Pass |  |
| 2 | User clicks profile | Transferred to profile page | | Pass |  |
| 3 | User clicks edit profile | Transferred to profile edit page | | Pass |  |
| 4 | User changes phone number |  | | Pass | **Allows users to change number to any number** |
| 5 | User clicks Save | User taken back to profile page | | Pass | **Users phone number is changed in the database** |
| Post-conditions:  N/A | | | | | |

### Test Case 24:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_24 | | | Test Case Name: Admins adding Admins | | |
| System: Control Panel | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: Admins change the member to admins | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | Admin clicks Login button | Login success | | Pass |  |
| 2 | Admin clicks Control Panel | Transferred to control panel page | | Pass |  |
| 3 | Admin clicks User Titles | Transferred to user titles page | | Pass |  |
| 4 | Admin enter a members name |  | | Pass |  |
| 5 | User clicks Search | Members with that name shows up | | Pass |  |
| 6 | Admin clicks Edit User |  | | Pass |  |
| 7 | Admin checks “Is user admin” box |  | | Pass |  |
| 8 | Admin clicks Save | User updated message shows up | | Pass | **Member is changed to admin in database** |
| Post-conditions:  N/A | | | | | |

### Test Case 25:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_25 | | | Test Case Name: Create Event | | |
| System: Control Panel | | | Subsystem: N/A | | |
| Designed By: Marquise Jennett | | | Designed Date: April 20, 2020 | | |
| Executed By: Marquise Jennett | | | Execution Date: April 20, 2020 | | |
| Short Description: The system shall allow admins to create events | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | Admin clicks Login button | Login success | | Pass |  |
| 2 | Admin clicks Control Panel | Transferred to control panel page | | Pass |  |
| 6 | Admin clicks Create Event | Transferred to create event page | | Pass |  |
| 7 | Admin enters Event title, location, date, and details |  | | Pass |  |
| 8 | Admin clicks post event | Confirmation message is displayed | | Pass |  |
| 9 | Admin clicks edit user |  | | Pass |  |
| Post-conditions:  Poll is added to database and displayed on Polls page | | | | | |

### Test Case 26:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_26 | | | Test Case Name: Login with incorrect credentials | | |
| System: Login Page | | | Subsystem: Sign Up Page | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: User logs in using non existing or incorrect credentials | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | User transferred to page with data fields that they are to fill in | | Pass |  |
| 2 | User email and name | Application prompts ‘name, email id doesn’t exist’ | | Pass |  |
| 3 | The page directs to the Signup page | User transferred to page with data fields that they are to fill in | | Pass |  |
| 4 | User clicks Submit button | If incorrect email or password error message is shown to the user | | Pass |  |
| Post-conditions:  See test cast 2 | | | | | |

### 

### 

### Test Case 27:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_27 | | | Test Case Name: Password length not between 8-16 | | |
| System: Login Page | | | Subsystem: Sign Up Page | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: User logs in with a password that has improper length | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | User clicks Login button | User transferred to page with data fields that they are to fill in | | Pass |  |
| 2 | User clicks sign up button | User transferred to sign up page | | Pass |  |
| 3 | User enter name, email, password, and phone number | Password length invalid text is shown | | Pass |  |
| 4 | User clicks ok | User is returned to sign up page | | Pass |  |
| Post-conditions:  See Test case 1 | | | | | |

### 

### Test Case 28:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_28 | | | Test Case Name: Admin Changing user titles | | |
| System: Alumni | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: Admin changing a users title | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | Admin logins | Successful Login Attempt  Transferred to home page | | Pass |  |
| 2 | Admin open menu bar | List of pages is displayed | | Pass |  |
| 3 | Admin clicks control panel | User transferred to control panel page | | Pass |  |
| 4 | Admin clicks user titles | User transferred to user titles page | | Pass |  |
| 5 | Admin enters first and last name of user |  | | Pass |  |
| 6 | Admin clicks Search | List of matching names show up | | Pass | **Names are pulled from Members database table** |
| 7 | Admin clicks edit user | User transferred to edit user page | | Pass |  |
| 8 | Admin enters new titles |  | | Pass |  |
| 9 | Admin clicks save | User is returned to user titles page | | Pass |  |
| Post-conditions:  Users title is updated in the database | | | | | |

### 

### Test Case 29:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case #: TC\_29 | | | Test Case Name:Admin Changing user titles (Search Failed) | | |
| System: Alumni | | | Subsystem: N/A | | |
| Designed By: Mansi Pandya | | | Designed Date: April 19, 2020 | | |
| Executed By: Mansi Pandya | | | Execution Date: April 19, 2020 | | |
| Short Description: Admin changing a users title using unrecognized name | | | | | |
| Step# | Action | Expected System Response | | Pass/ Fail | Com-ment |
| 1 | Admin logins | Successful Login Attempt  Transferred to home page | | Pass |  |
| 2 | Admin open menu bar | List of pages is displayed | | Pass |  |
| 3 | Admin clicks control panel | User transferred to control panel page | | Pass |  |
| 4 | Admin clicks user titles | User transferred to user titles page | | Pass |  |
| 5 | Admin enters first and last name of user |  | | Pass |  |
| 6 | Admin clicks Search | Name cannot be found error message pops up  Returned to user title page | | Pass | **Names are pulled from Members database table** |
| Post-conditions:  See test case 17 and/or 21 | | | | | |

## Test coverage(Statement/branch):

### MemberListViewModel, GetMembers():

Number of executed statements = 32

Total number of statements = 42

Statement coverage = (32/42) x 100 = 76.2%

Considering the most minimum branch, number of branches covered = 1

Total branches/decision = 4

Branch coverage= (1/4) x 100 = 25%

### MemberListViewModel, GetMentors():

Number of executed statements = 8

Total number of statements = 11

Statement coverage = (8/11) x 100 = 72.7%

Considering the most minimum branch, number of branches covered = 1

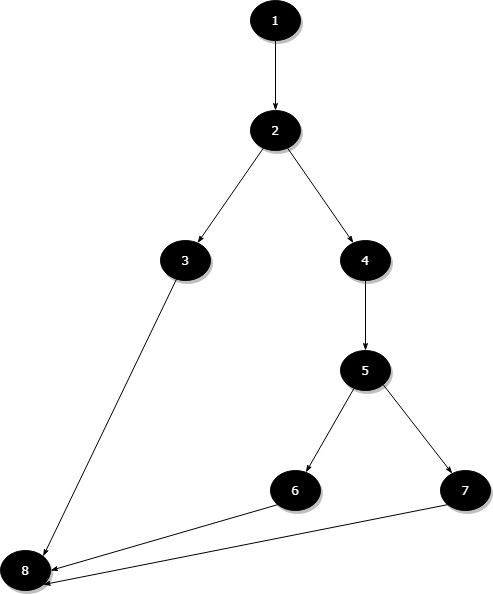
Total branches/decision = 2

Branch coverage= (1/2) x 100 = 50%

## 

## Product/Process quality (Mccabe’s complexity):

### MemberListViewModel, GetMembers():

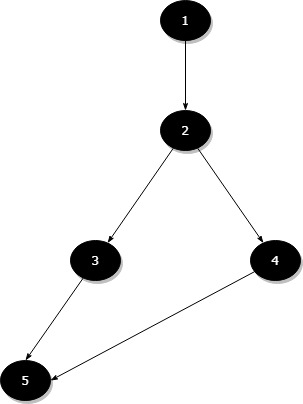


Mathematically, Mccabe’s number is, V(G) = E - N + 2

Where, E - Number of edges, N - Number of Nodes.

Mccabe’s number = 9-8+2= 3

### MemberListViewModel, GetMentors():



Mathematically, Mccabe’s number is, V(G) = E - N + 2

Where, E - Number of edges, N - Number of Nodes.

Mccabe’s number = 5-5+2= 2