**Data Dictionary**

|  |  |
| --- | --- |
| **Class Name:** User | **ID:** 1 |
| **Superclasses:** None | |
| **Attributes:**   * userName. String type. Username of the user * firstName. String type. * lastName. String type. * email. String type. * password. String type * university. String type. * expertise. String type. Words or phrases describing the expertise of the user. * type. String type. User type. The role the user is holding. * numPaperAssigned. Int type. Number of paper assigned to the user * paperAssigned. Int type. The ID of the paper assigned to the user | |
| **Methods:**   * setAll()   Set all the attributes in the class   * setUserName()   Set the username of the user   * setFirstName()   Set the first name of the user   * setLastName()   Set the last name of the user   * setEmail()   Set the email of the user   * setUniversity()   Set the university of the user   * setExpertise()   Set the expertise of the user   * setPassword()   Set the password of the user   * setType()   Set the type of the user.   * getUsername()   Get the username of the user   * getPassword()   Get the password of the user   * getType()   Get the type of the user   * getEmail()   Get the email of the user   * getExpertise()   Get the expertise of the user   * getNumPaperAssigned()   Get the number of paper assigned to the user   * getPaperAssigned()   Get the list of paper assigned to the user   * addNumPaperAssigned()   Add the number of paper assigned to the user   * addPaperAssigned()   Add in the paper ID of the recently assigned paper   * display()   Display details of the user | |

|  |  |
| --- | --- |
| **Class Name:** UserControl | **ID:** 2 |
| **Superclasses:** None | |
| **Attributes:**  None (This class is used to check details passed in from the UserBoundary) | |
| **Methods:**   * registerUser()   Function to register user, obtain data from boundary classes   * login()   Function to check login details passed in by the boundary   * checkUN()   Check the validity of the username   * checkPW()   Check the validity of the password   * changePassword()   Change the password for the user   * updateEmail()   Update the email of the user   * updateUniversity()   Update the university of the user   * updateExpertise()   Update the expertise of the user   * count()   Count the number of records available for the user | |

|  |  |
| --- | --- |
| **Class Name:** UserBoundary | **ID:** 3 |
| **Superclasses:** None | |
| **Attributes:**   * userControl. UserControl type. | |
| **Methods:**   * registerUSer()   Obtain input details from the user for registering   * login()   Obtain login details from the user   * changePassword()   Obtain details from the user to change password   * changeDetails()   Change the details of the user | |

|  |  |
| --- | --- |
| **Class Name:** SystemMain | **ID:** 4 |
| **Superclasses:** None | |
| **Attributes:**   * currentlyLoggedInUser. String type. Keeps track of the currently logged in user. * userType. String type. Keep track of the type of the user | |
| **Methods:**   * mainPage()   The page where user login or registers.   * homePage()   The page that directs the currently logged in user to their homepages based on their type   * adminPage()   The page for the admin. Each page have different layouts   * authorPage()   The page for the author.   * PCpage()   The page for the program committee.   * PCChairPage()   The page for the program committee chair   * loadFunctionalities() * The function to load the latest functionalities (used to check if there is a change in the state of functionalities e.g submission of paper, submission of reviews). Refreshed every time it returns to the menu. | |

|  |  |
| --- | --- |
| **Class Name:** Author | **ID:** 5 |
| **Superclasses:** None | |
| **Attributes:**  None (This class is used to call functions from other classes) | |
| **Methods:**   * changeDetails()   Calls the modify details function from the UserBoundary   * changePassword()   Calls the modify password function from UserBoundary   * submitPaper()   Calls the submit paper function from PaperManagement   * modifyPaperSubmission()   Calls the modify paper submission function from PaperManagament   * notifications()   Calls the notification function from the PaperManagement to return the status of the paper, either rejected or approved   * participateConference()   Participates in a conference available   * countConference()   Count the number of recorded conference | |

|  |  |
| --- | --- |
| **Class Name:** ProgramCommittee | **ID:** 6 |
| **Superclasses:** Author | |
| **Attributes:**  None (This class is used to call functions from other classes. Functions mostly inherited from previous classes.) | |
| **Methods:**   * specifyPreference()   Calls the function to specify preference from preference management   * reviewPaper()   Calls the function to review paper from PaperManagement   * modifyReview()   Calls the function to modify review paper from PaperManagement   * reviewDiscussion()   Calls the function to start the discussion among reviewers from PaperManagement | |

|  |  |
| --- | --- |
| **Class Name:** PCChair | **ID:** 7 |
| **Superclasses:** Author, ProgramCommittee | |
| **Attributes:**  None (This class is used to call functions from other classes. Functions mostly inherited from previous classes) | |
| **Methods:**   * assignPC()   Function to assign program committee. Can only be done to those having the role of authors   * monitorPC()   Calls the function to monitor the program committee from the PaperManagement   * functionalityManagement()   Calls the menu function to change functionalities in the system from FunctionalitiesManagement   * countUser()   Function that counts the number of user in the system   * writeFile()   Function that updates the user file after updating it with the latest PC member   * loadFile()   Function to load the data from the user file   * latestEvents()   Function that displays the latest events of the system. New user registration, submitting a paper and paper reviews   * approvePaper()   Calls the function to approve paper from PaperManagement   * manuallyAssignPaper()   Calls the function to manually assign paper from PaperManagement | |

|  |  |
| --- | --- |
| **Class Name:** Admin | **ID:** 8 |
| **Superclasses:** Author, ProgramCommittee, PCChair | |
| **Attributes:**  None (Functions mostly inherited from previous clases. Class used to call function from other classes) | |
| **Methods:**   * assignPCChair()   Function that assigned the role of PCChair to users.   * generateConference()   Generates a new conference | |

|  |  |
| --- | --- |
| **Class Name:** Conference | **ID:** 9 |
| **Superclasses:** None | |
| **Attributes:**   * conferenceName. String type. The name of the conference * location. String type. The location of the conference * numUSer. Int type. The number of user participated in the conference * userList. Vector of int. The list of users who participated in the system | |
| **Methods:**   * addNumUser()   Function to add the number of user participating in the conference   * addUser() * Function to add the user ID into the conference * setLocation()   Function to set the location for the conference   * setConferenceName()   Function set the conference name   * getConferenceName()   Function to get the name of the conference | |

|  |  |
| --- | --- |
| **Class Name:** PreferenceManagement | **ID:** 10 |
| **Superclasses:** None | |
| **Attributes:**  None (This class mostly loads data from the database and manipulate accordingly to the need) | |
| **Methods:**   * specifyMaybe()   Function that specifies preference of “Maybe” to all program committees who have yet to specify preference for the papers.   * specifyPreference()   Function that specifies preference for all papers in the system for the currently logged in user   * checkUser()   Function to check if the user has already submitted his preferences for all the papers before. IF the user has submitted a preference before, we would need to update the whole file again, else we just add it to the end of the preference file   * countPreference()   Count the number of users who have contributed their preference so far   * countUser()   Counts the number of user   * countPaper()   Counts the number of papers in the system   * writeApp()   Updates preference to the end of the file instead of rewriting all of them   * writeAll()   Rewrites the preferences into the file. | |

|  |  |
| --- | --- |
| **Class Name:** Preference | **ID:** 11 |
| **Superclasses:** None | |
| **Attributes:**   * userName. The username of the user who specified his/her preferences on papers. * preference. Vector of int. As the preferences are only done after paper submission is done, the vector position will determine which paper it is based on the sequence of the list of papers submitted in the system. Size of the vector would be the number of paper in the system.   (e.g Submitted papers – 1001, 1002 | vector position of 1 for preference would be the preference for paper 1001, vector position of 2 would be the preference for paper 1002.) | |
| **Methods:**   * addPreference()   Pushes the preference of the user for the paper into the vector   * setUsername()   Sets the username to the currently logged in user, who is specifying preferences for the paper   * getUsername()   Get the username of the user.   * countPaper()   Count the number of papers in the system   * getPreference()   Get the preference for the paper of the user based on the int fed in the function   * getAllPreference()   Returns the whole vector of paper preferences | |

|  |  |
| --- | --- |
| **Class Name:** md5 | **ID:** 12 |
| **Superclasses:** None | |
| **Attributes:**   * (This function is based on code from the internet. It is used to encrypt the password.   Obtained from http://www.zedwood.com/article/cpp-md5-function) | |
| **Methods:** | |

|  |  |
| --- | --- |
| **Class Name:** FunctionalitiesManagement | **ID:** 13 |
| **Superclasses:** None | |
| **Attributes:**  None. (This class loads data from database and manipulate accordingly to the need of the user) | |
| **Methods:**   * menu()   The menu that will display the functionalities available for changing   * generateConferenceProceeding()   Function that takes the list of approved paper and generated the conference proceeding   * autoSpecifyPreference()   Calls the specifyMaybe() function from PreferenceManagement   * setReviewerPaperReceive()   Set the number of reviewers each paper will receive. Default set at 5   * setPaperReviewerReceive()   Set the number of papers each reviewer will receive. Default set at 3   * enableRevSubmission()   Enable submission of reviews   * enablePapSubmission()   Enable submission of papers   * enableAutResponse()   Enable author response (NOT USED)   * enableRevDiscussion()   Enable discussion among reviewers about reviews done on assigned paper   * autoAssignPaperToReviewers()   Function that automatically assigns paper to reviewers. Multiple checking done in this function to make sure paper is not assigned to contributors and much more.   * countPaper()   Count the number of paper in the system   * countUser()   Count the number of the user in the system   * countPreference()   Count the number of preferences in the system   * loadFunctionalities()   Load the latest functionalities into the system   * writeFunctionalities()   Updates the functionalities text file with the latest changes   * stringSplit()   This is used for auto assigning paper. As user will be prompted to key in their expertise/keywords by splitting them using “.”. This function splits the string apart   * writeUser()   Updates the user text file. | |

|  |  |
| --- | --- |
| **Class Name:** Functionalities | **ID:** 14 |
| **Superclasses:** None | |
| **Attributes:**   * reviewDiscussion. Integer type. -1 means discussion among reviewers are allowed, else not allowed. * reviewSubmission. Integer type. -1 means reviews cannot be submitted by reviewers, else not allowed * authorResponse. Integer type. -1 means authors cannot leave a response, else not allowed. (Functions to take in author response NOT IMPLEMENTED) * paperSubmission. Integer type. -1 means authors cannot submit a paper, else not allowed. * paperReviewerReceive. Integer type. The number of papers reviewers can receive * reviewerPaperReceive. Integer type. The number of reviewer papers can receive | |
| **Methods:**   * setReviewDiscussion()   Change the value for reviewDiscussion   * setReviewSubmission()   Change the value for reviewSubmission   * setPaperSubmission()   Change the value for paperSubmission   * setAuthorResponse()   Change the value for author response   * setPaperReviewerReceive()   Set the number of papers reviewers receive   * setReviewerPaperReceive()   Set the number of reviewers paper receive   * getReviewDiscussion()   Returns reviewDiscussion   * getReviewSubmission()   Gets the value for reviewSubmission   * getAuthorResponse()   Gets the value for author response   * getPaperSubmission()   Gets the value for paper submission   * getPaperReviewerReceive()   Get the number of paper reviewers receive   * getReviewersPaperReceive()   Get the number of reviewers paper receive | |

|  |  |
| --- | --- |
| **Class Name:** PaperManagement | **ID:** 15 |
| **Superclasses:** None | |
| **Attributes:**  None (Class that loads data from the database and manipulates accordingly.) | |
| **Methods:**   * submitPaper()   Function used to submit paper   * modifyPaperSubmission()   Function that modifies paper submission   * reviwPaper()   Function that is used to review papers assigned to the currently logged in user. Can’t review own paper   * modifyReview()   Function that modifies review done by the currently logged in user   * writeFile()   Write a paper record into the file   * writeAll()   Rewrites the whole paper text file   * writeAssignment()   Updates the file with the paper assignment.   * transferFile()   Transfers file from user directory to the papers folder   * reverseTransfer()   Transfer from papers folder to users current directory. For user to read the paper he is reviewing   * generateID()   Generates an ID for the paper   * countPaper()   Counts the number of paper in the system   * countUser()   Counts the user in the system   * countPreference()   Counts the number of preferences by users   * makeReview()   Generates a new review for the paper   * reviewDiscussion()   Discussion page for reviewers   * countDiscussion()   Count the number of discussions in file   * submitAuthorResponse()   (NOT IMPLEMENTED)   * approvePaper()   Function to approve or reject paper   * manuallyAssignPaper()   Function to manually assign paper to user   * notifications()   Tells the user if his/her paper is accepted or rejected   * monitorPC()   Monitors PC activities | |

|  |  |
| --- | --- |
| **Class Name:** ResearchPaper | **ID:** 16 |
| **Superclasses:** None | |
| **Attributes:**   * paperID. Integer type. The ID of the paper * numContributors. Integer type. The number of contributor for the paper * contributedFirst. Vector of string. The list of contributed user’s first name * contributedLast. Vector of string. The list of contributed user’s last name * cntributedEmail. Vector of string. The list of contributed user’s email/ * contributedUni. Vector of string. The list of contributed user’s university. * title. String type. The title of the paper * abstract. String type. The abstract of the paper * keywords. String type. The keywords of the paper * approval. Int type. Approval of the paper. -1 for rejected, 1 for approved | |
| **Methods:**   * getTitle()   Get the title of the paper.   * setPaperID()   Set the ID of the paper   * setNumContributors()   Set the number of contributors for the paper   * setContributedLast()   Set the last name of the contributors   * setContributedFirst()   Set the first name of the contributors   * setContributedUni()   Set the university of the contributors   * setContributedEmail()   Set the email of the contributors   * setTitle()   Set the title of the paper   * setAbstract()   Set the abstract of the paper   * setKeywords()   Set the keywords of the paper   * getPaperID()   Get the ID of the paper   * getApproval()   Get the approval of the paper   * getKeywords()   Get the keywords of the paper   * getContributedEmail()   Get the contributed emails of this paper   * getNumContributors()   Get the number of contributors for the paper   * setApproval()   Set the paper to approved   * setReject()   Set the paper to rejected   * checkEmail()   Feed in an email and it will check if the email exist among the contributors   * display()   Display details about the paper | |

|  |  |
| --- | --- |
| **Class Name:** PaperReview | **ID:** 17 |
| **Superclasses:** None | |
| **Attributes:**   * reviewedBy. String type. The username of the user who reviewed the paper * strengths. String type. The strengths of the paper * weakness. String type. The weakness of the paper * comments. String type. The comments on the paper by the user * suitability. Int type. The suitability of the paper * pcRemarks. String type, The PC’s remarks on the paper * overall. Int type. The overall of the paper * reviewerConfidence. Int type. The confidence of the reviewer * relevance. Int type. The degree of relevance to the conference * originality. Int type. The originality of the paper * significance. Int type. The significance of the paper * presentation. Int type. The presentation of the paper. * technicalQuality. Int type. The technical quality of the paper * evaluation. Int type. The evaluation of the paper | |
| **Methods:**   * getReviewedBy()   Get the user who reviewed the paper   * setReviewedBy()   Set the user who reviewed the paper   * setStrengths()   Set the strengths of the paper   * setWeakness()   Set the weakness of the paper   * setComments()   Set the comments of the paper   * setSuitability()   Set the suitability of the paper   * setPcRemarks()   Set the PC’s remarks on the paper   * setOverall()   Set the overall of the paper   * setReviewerConfidence()   Set the reviewer confidence on the paper   * setRelevance()   Set the relevance of the paper   * setOriginality()   Set the originality of the paper   * setSignificance()   Set the significance of the paper   * setPresentation()   Set the presentation of the paper   * setTechnicalQuality()   Set the technical quality of the paper   * setEvaluation()   Set the evaluation of the paper   * display()   Display information the review done on the paper   * editReview()   Modify attributes of the paper | |

|  |  |
| --- | --- |
| **Class Name:** PaperDiscussion | **ID:** 18 |
| **Superclasses:** None | |
| **Attributes:**   * paperID. Integer type. The discussion for the paper * numReviewerResponse. Int type. The number of reviewer response on the paper * reviewerDiscussion. Vector of string. The reviewer response on the papers * numUserResponse. Int type. The number of user response on the paper (NOT IMPLEMENTED) * userResponse. Vector of string. The user response on the paper | |
| **Methods:**   * setPaperID()   Set the paper ID for the discussion   * addNumReviewerResponse()   Add the number of reviewer response   * addReviewerDiscussion()   Add the reviewer discussion   * addNumUserResponse()   Add the number of user response (NOT IMPLEMENTED)   * addUserResponse()   Add the user response (NOT IMPLEMENTED)   * getPaperID()   Get the ID of the paper this discussion is for   * getNumReviewerResponse()   Get the number of reviewer response   * returnReviewerDiscussions()   Get the list of reviewer responses   * returnUSerResponse()   Get the list of user responses (NOT IMPLEMENTED) | |

|  |  |
| --- | --- |
| **Class Name:** PaperAssignment | **ID:** 19 |
| **Superclasses:** None | |
| **Attributes:**   * paperID. Int type. The ID of the paper * numAssignedForReview. Int type. The number of reviewers assigned for the paper * userList. Vector of string. The list of usernames assigned to the paper * numUserReviewed. The number of user who has reviewed the paper * userReview Vector of class PaperReview\*, The reviews done by the users. | |
| **Methods:**   * setPAperID()   Sets the ID of the paper   * setNumAssignedForReview()   Set the number of people assigned to review this paper   * addUser()   Adds a new user to the list of users assigned for the paper   * addNumUserReviewed()   Adds the number of user who has reviewed the paper   * addUserReview()   Adds the user review   * getNumAssingedForReview()   Get the number of people assigned to review this paper   * getUserList()   Get the list of users reviewing this paper   * getNumUserReviewed()   Get the number of user who has reviewed the paper   * getUserReview()   Get the user reviews   * getPAperID()   Get the ID of the paper | |