

**Computer Science and Engineering**

**Integrated University Department Information System**

**Requirements and Analysis Specification**

**Version 1.0**

Document Number: RAS-001

Team Number: **B6**

Team Members: **Munieshwar Ramdass (mr3420)**

**Ajay Shenoy (as7195)**

**August Tan (aot221)**

**Review and Approvals**

|  |  |  |  |
| --- | --- | --- | --- |
| **Printed Name and Title** | **Function** | **Date** | **Signature** |
| Professor Strauss | Advisor, Project Manager | February 20, 2015 | On File |
| Munieshwar Ramdass | Programmer, Author, Reviewer | February 20, 2015 | On File |
| Ajay Shenoy | Programmer, Author, Reviewer | February 20, 2015 | On File |
| August Tan | Programmer, Author, Reviewer | February 20, 2015 | On File |

**Revision Level**

|  |  |  |
| --- | --- | --- |
| **Date** | **Revision Number** | **Purpose** |
| February 20, 2015 | Version 1.0 | Initial Release |

**Table of Contents**

**1. INTRODUCTION ................................................................................................................... 6**

1.1 PURPOSE .................................................................................................................... 6

**2. SCOPE ...................................................................................................................................... 6**

2.1 IDENTIFICATION .............................................................................................……. 6

2.2 BOUNDS ..................................................................................................................... 6

2.3 OBJECTIVES .............................................................................................................. 6

2.4 SYSTEM OVERVIEW ............................................................................................... 7

2.5 DOCUMENT OVERVIEW ......................................................................................... 7

**3. REFERENCE DOCUMENTS ................................................................................................ 7**

**4. BUSINESS REQUIREMENTS ............................................................................................ 7**

4.1 TECHNOLOGY ........................................................................................................ 7

4.2 ECONOMICS ............................................................................................................ 7

4.3 REGULATORY AND LEGAL ................................................................................. 7

4.4 MARKET CONSIDERATIONS ............................................................................... 7

4.5 RISKS AND ALTERNATIVES .................................................................................8

4.6 HUMAN RESOURCES AND TRAINING ................................................................8

**5. DESCRIPTIVE FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS ……. 8**

5.1 SYSTEM’S PURPOSE …………………………………………………………….... 8

5.2 FUNCTIONAL DESCRIPTIVE DETAILED REQUIREMENTS …………………. 9

5.3 NON-FUNCTIONAL DESCRIPTIVE DETAILED REQUIREMENTS …………. 10

5.4 CONTEXT DIAGRAM …………………………………………………………. 10

**6. LOGICAL ARCHITECTURAL SPECIFICATION ........................................................ 11**

6.1 SYSTEM CAPABILITY REQUIREMENTS .......................................................... 11

*6.2.1 Capabilities .................................................................................................11*

6.2 USER INTERFACE REQUIREMENTS .................................................................. 16

6.3 COMPONENT (COMPONENT/PACKAGE/SUBSYSTEM) ARCHITECTURE .. 17

*6.3.1 Component Descriptions ............................................................................. 17*

*6.3.2 Component Architecture Diagram .............................................................. 17*

6.4 CLASS DIAGRAMS ................................................................................................. 18

6.5 CLASS RELATIONSHIP/INTERACTION DIAGRAMS ....................................... 19

6.6 EVENT SECTION ..................................................................................................... 20

*6.6.1 Event Dictionary ......................................................................................... 20*

*6.6.2 Event Diagrams .......................................................................................... 21*

6.7 ACTIVITY/STATE (SCENARIO) SECTION .......................................................... 21

*6.7.1 Activity (Scenario) Diagrams ……………………………….…………………. 21*

*6.7.2 Activity (Scenario) Specification …………………………....……………….... 22*

6.8 SEQUENCE DIAGRAMS ........................................................................................ 22

6.9 COLLABORATION DIAGRAMS ........................................................................... 26

6.10 DICTIONARIES ...................................................................................................... 26

**7. NON-FUNCTIONAL/OPERATIONAL SPECIFICATIONS .......................................... 33**

7.1 SYSTEM EXTERNAL INTERFACE REQUIREMENTS ....................................... 33

7.2 SAFETY REQUIREMENTS .................................................................................... 33

7.3 SECURITY AND PRIVACY REQUIREMENTS .................................................... 34

7.4 SYSTEM ENVIRONMENT REQUIREMENTS ...................................................... 34

7.5 COMPUTER RESOURCE REQUIREMENTS ........................................................ 34

*7.5.1 Computer Hardware Requirements ............................................................ 34*

*7.5.2 Computer Hardware Resource Requirements ............................................ 34*

*7.5.3 Computer Software Requirements .............................................................. 34*

*7.5.4 Computer Communications Requirements ................................................. 34*

7.6 SYSTEM QUALITY FACTORS .............................................................................. 34

7.7 DESIGN AND CONSTRUCTION CONSTRAINTS ............................................... 34

7.8 PERSONNEL-RELATED REQUIREMENTS ......................................................... 34

7.9 TRAINING-RELATED REQUIREMENTS ............................................................. 34

7.10 LOGISTICS-RELATED REQUIREMENTS .......................................................... 35

7.11 PACKAGING REQUIREMENTS .......................................................................... 35

7.12 PRECEDENCE AND CRITICALITY REQUIREMENTS .................................... 35

7.13 OTHER REQUIREMENTS .................................................................................... 35

**8. SYSTEM TEST PLAN REQUIREMENTS ........................................................................ 35**

**9. QUALIFICATION PROVISIONS ...................................................................................... 35**

**10. REQUIREMENTS TRACEABILITY ............................................................................... 35**

**11. RATIONALE ....................................................................................................................... 36**

**12. NOTES .................................................................................................................................. 36**

**13. APPENDICES ...................................................................................................................... 36**

13.1 DICTIONARIES ...................................................................................................... 36

13.2 UML DIAGRAMS ………...…………………………………….……………….. 36

13.3 SCHEDULE TRACKING ....................................................................................... 36

13.4 DEFECT TRACKING ............................................................................................. 37

**1. INTRODUCTION**

1.1 PURPOSE

This Requirements and Analysis Specification (RAS) will be given to the Software Assurance group (SQA), the development team, and the client. The SQA group will use this document as a means to tests the product through its life cycle while the development team uses the Software Project Management Plan (SPMP) to aid in implementing their product. The client will confirm that the his/her demands are met.

**2. SCOPE**

2.1 IDENTIFICATION

Integrated University Department Information System, B6, RAS Version 1.0, 2/20/2014

2.2 BOUNDS

The Integrated University Department Information System is expected to be used by university administrators, who will have complete access, and professors and students who will have limited access to the system. The university’s system may interact with the IUDIS as necessary. Unique to administrators, they can add and delete users. Professors may submit/change students’ grade and request lab materials and other items or personnel which may alter the budget of the university, as a result of extensive research and testing done by PHD student. Professor’s salary will be taken to account. Students may view grades and financial information. Students may make payments to their tuition and lab fees. Third party companies may access the IUDIS if they are a mundane part of the system in which they continuously provide goods/services, therefore building a tab for the university to monitor. The IUDIS will feature a departmental registrar. Ultimately, the IUDIS will only facilitate users of a specific department in a specific university and third party goods/services.

2.3 OBJECTIVES

Project Priority: The Integrated University Departmental Information System is an organized approach intended to help universities regulate student records, laboratory administration, the ordering of goods and services, and the management of account receivable and payable. It should be linked to a wider University system responsible for staff salaries, student transcripts, etc. The IUDIS structure provides a new information system that is faster with a slicker user interface then the outdated and archaic systems, which makes it easier for students to access their records. Simultaneously, it will be effortless for staff and faculty to use the system for submitting grades and financial information. The new user interface allows for users to interact with the system seamlessly while being aesthetically pleasing, accommodating a broader spectrum of users. The structure of the information system will be rigorously secure to protect the confidentiality, integrity and availability of the individual user.

Life Cycle: Unified Process

Deliverables: Due Date:

Project Team Selection Form February 6, 2015

Project Proposal February 13, 2015

Requirements and Analysis Specification (RAS) February 27, 2015

Software Project Management Plan (SPMP) March 6, 2015

Software Design Description (SDD) - Initial March 20, 2015

Software Design Description (SDD) - Final April 17, 2015

Implementation/Demonstration April 24, 2015 - May 4, 2015

Formal Oral Presentation April 24, 2015 - May 4, 2015

2.4 SYSTEM OVERVIEW

The Integrated University Departmental Information System is a smaller part of the larger university system that will interact with the budget system and the database system for the computer science department. The IUDIS is concerned with the users of the IUDIS, such as students, professors, advisors, other officials, and third party goods/service providers. The goal of the IUDIS is to please the user by having fast loading, easy to read, clearly arranged, and seamless navigation to all pages. Other factors that may affect the system is the load, the hard disk space, and the amount of down time the server faces. This product may be appear to be a combination of NYU’s Albert and Bursar systems.

2.5 DOCUMENT OVERVIEW

This Software Project Management Plan (SPMP) will describe the business requirements while revisiting the functional and nonfunctional requirements from the Software Requirements Specifications (SRS). There are also details on requirements traceability and document evolution over this project’s lifecycle. Furthermore, this document will keep track of any progress in development.

**3. REFERENCE DOCUMENTS**

Integrated University Department Information System, A6, Project Proposal Version 1.1

Integrated University Department Information System, A6, SRS Version 1.1

Integrated University Department Information System, A6, SPMP Version 1.0

Integrated University Department Information System, A6, SAS Version 1.0

Integrated University Department Information System, B6, Project Proposal Version 1.2

**4. BUSINESS REQUIREMENTS**

4.1 TECHNOLOGY

We will be using technology to make managing school grades and tuitions easier. It will be simplified and faster as a result.

4.2 ECONOMICS

This project is funded by the universities that have chosen to purchase a subscription. The cost of the product will depend on the features that the client desires to include in the overall product. Project will use as much open source materials as possible to minimize cost.

4.3 REGULATORY AND LEGAL

Regulatory and Legal requirements will not be discussed.

4.4 MARKET CONSIDERATIONS

There will be no market drivers for this system. This system is made for very populated universities and these universities may choose to purchase the rights via subscription if it works with their larger university system.

4.5 RISKS AND ALTERNATIVES

Unexpected functionality may occur. To prevent this, administrators must rigorously test the system for flaws and fix it throughout the system’s lifecycle. Since this system can be accessed through the internet, hackers might be able to hack the database. To prevent this, a firewall and/or an intranet can be use.

4.6 HUMAN RESOURCES AND TRAINING

Priority is placed heavily on training individuals in object oriented design. This is especially needed for information hiding. All members must have knowledge on Databases. A course in Databases is necessary. All members must have intermediate coding skills in Java which can be obtained by taking a course. All members must be familiar with Maven which can be attained by text and/or online resources. All members must have familiarity with web applications. This includes familiarity with HTML5/CSS3 which can be learned through text and/or online resources.

**5. DESCRIPTIVE FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS**

5.1 SYSTEM’S PURPOSE

The purpose of this system is to allow the different user entities to communicate, access official school records, and enroll in the various universities offered courses. Also, a departmental budget is being kept, which will be tallied to the university system.

5.2 FUNCTIONAL DESCRIPTIVE DETAILED REQUIREMENTS

There must be a method to handle the Management of Registrar.

1. Administrators/Officials must be able to create/modify/delete/view class.

1.1. Professors may vote for a prerequisite for a class.

1.2. Professors may vote on credit level for a class.

1.3. Officials must be able to set time for class.

1.4. Professors may to vote and associate a lab and/or recitation to class.

2. Administrators/Officials must be able to create/modify/delete/view any user account.

2.1. Administrators must be able to assign user permissions.

2.2. Administrators must have absolute control.

2.3. University’s system’s administrators must have absolute control.

2.4. Hold can depend on user financial status.

2.5. Hold can depend on student prerequisite.

2.6. Hold can depend on other predefined reasons by officials.

2.7. Professor must be able request to teach a class.

2.8. Students must be able request to take a class.

2.9. Professors must be able post grade for students in given class(es).

3. There must be a way to update student transcript and public them.

3.1. Student must be able to request Unofficial Transcript.

3.2. Student must be able to request Official Transcript.

3.3. Officials must be able to sign Official Transcript.

3.4. Grades and courses must be reflected on the student’s transcript.

There must be a method to handle the Management of Finance.

4. Officials from university system must be able to handle all expenses.

4.1. Professors must be able to request research funds.

4.2. Officials must be able to approve/disapprove professor’s request for fund.

4.3. Professors must be able to request goods and services.

4.4. Officials must be able to approve/disapprove professor request.

5. There must be a payable and receivable system.

5.1. Officials may make payments.

5.2. University’s system’s officials may handle payments.

5.3. Students must be able make payments via their means of paying.

5.4. Officials must be able allow third party company to allow student scholarship/grant/payments.

5.5. A direct deposit system may be used from university system.

5.6. A check mailing system may be used if direct deposit not used.

There must be a method of Communication.

6. The use of the integrated university’s emailing and mailing system will be used.

6.1. Emails will primarily be notifications from user account changes.

6.2. Officials may choose to quickly send an email from user interface to the person of whom s/he is currently viewing.

There must be a method to handle the Maintenance of IUDIS.

7. Administrators must have access to entire IUDIS and its database.

7.1. Administrators must be able to debug the entire system.

7.2. Integrated permissions depend on larger university system.

Credentials

8. There must be a way for users to login and logout.

8.1 User enters username and password to login.

8.2 User requests to logout if signed in.

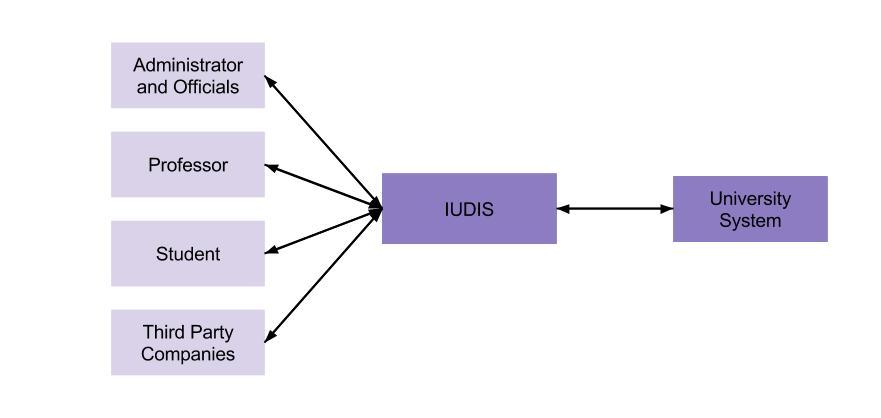
5.3 NON-FUNCTIONAL DESCRIPTIVE DETAILED REQUIREMENTS

A student should not be given the permissions (and interface) of the professor, official, and administrator. A professor should not be given the permissions (and interface) of the student, official, and administrator. A official should not be given the permissions (and interface) of the student, professor, and administrator. This system must manage about 50,000 students and 5,000 non-students. For actual technical specifications refer to section 7.3 of this document.

IUDIS will provide a University with a system to manage their registrar, finance, communication, and maintenance. These services include student tuition, student grades, faculty salaries, and other expenses. The IUDIS will provide different utilities depending on the user’s class. For instance, an Administrator will be able to create new accounts, remove accounts, modify accounts, manage academic records, create/modify classes, and manage salaries, tuition payments, and any other expenses.

Meanwhile, a Professor will have some limitations compared to an Administrator. He or she will be assigned a class by an Official. He can also post student grades for his class. Lastly, a professor will be able to request funds for research.

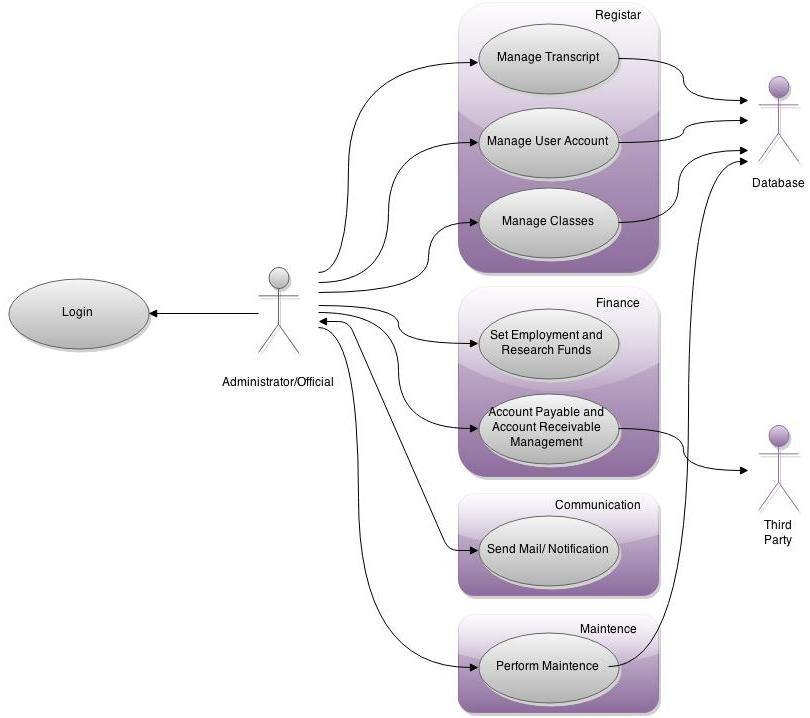
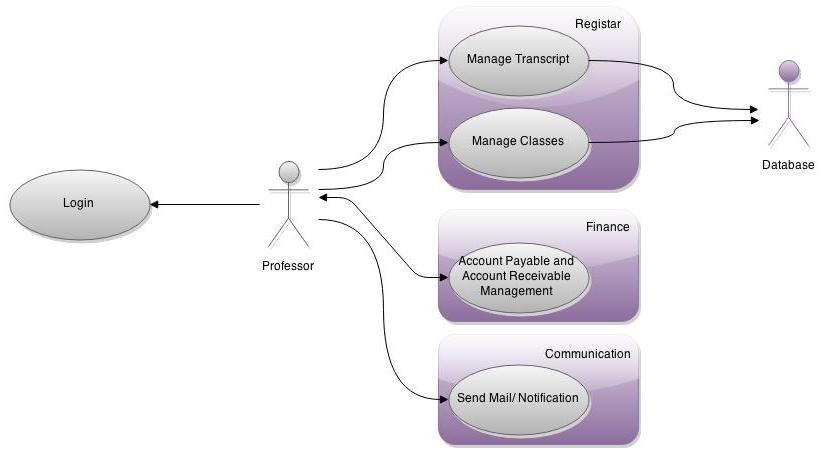
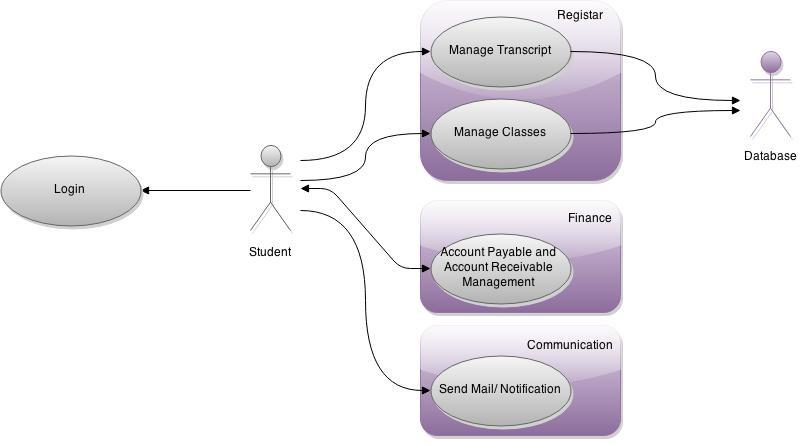
5.4 CONTEXT DIAGRAM



**6. LOGICAL ARCHITECTURAL SPECIFICATION**

6.1 SYSTEM CAPABILITY REQUIREMENTS

*6.1.1 Capabilities*



|  |  |  |
| --- | --- | --- |
| Login | | |
| Description | Admins, Officials, Professors and Students can use their credentials to log into the system with permissions suited based on their status. Incorrect credentials does not allow access. | |
| Pre-Conditions | User must be a part of university’s department that uses this software. User must have their account created by admin/official. | |
| Flows | Basic or Normal Flows | 1 User enters username and password.  2 System checks database for correct login credentials.  3 User now have access to their account. |
|  | Alternative Flows | 1 From step 1 above, if user enters wrong credentials, prompt error. Use case ends.  2 Use case ends if user quits/log out. |
| Post Conditions | Depending on who the user is, they will be able to make different transactions within the IUDIS. | |
| Special Requirements | Only one log in per device permitted. | |
| Extension Points |  | |

|  |  |  |
| --- | --- | --- |
| Manage Student Transcript | | |
| Description | Based on person that is logged into the system, user will either be able to edit and/or view transcript. | |
| Pre-Conditions | User must be logged in. | |
| Flows | Basic or Normal Flows | 1 Officials make and view students’ transcript.  2 Professor will then be able to edit grades.  3 Students will lastly be able to view grades on a transcript. |
|  | Alternative Flows | 1 Student may request a copy.  2 Transcript not available: present nothing to show or edit until Official makes it for student. |
| Post Conditions | There must always be an existing student transcript stored in the database, even if the student is no longer in the University. | |
| Special Requirements |  | |
| Extension Points |  | |

|  |  |  |
| --- | --- | --- |
| Management of Classes | | |
| Description | This is how the registrar’s classes are created and filled with students and associated to a professor that will teach the class. Based on user permissions, user will be able to do different things. | |
| Pre-Conditions | User must be logged in. | |
| Flows | Basic or Normal Flows | 1 Official creates (opens) a class  2 Professor may request teaching it  3 Professor may vote on class’s credit level and prerequisites.  4 Student may choose to register for class or remove themselves from class. |
|  | Alternative Flows | 1 Block student from registering for class if s/he does not have prerequisites.  2 Block professor from teaching if s/he is not fit to teach it.  3 Block student if his/her financial side is not taken care of; create an account hold in the Management of User Accounts used case. |
| Post Conditions | Either student is registered for classes or professor given a class. | |
| Special Requirements |  | |
| Extension Points |  | |

|  |  |  |
| --- | --- | --- |
| Management of User Accounts | | |
| Description | This is how administrators/officials will create and edit user accounts. Students and Professors will be able to view their account. | |
| Pre-Conditions | User must be logged in. | |
| Flows | Basic or Normal Flows | 1 Admins/officials can create user account, which user will be able to use in the department.  2 Professor will be able to view their account information  3 Student will be able to view their account information |
|  | Alternative Flows | 1 Account hold created if user’s financial side has not been taken care of. Access will be blocked to registrar until they handle their financial side in the Account Receivable and Payment Management used case.  2 Students may be restricted to prerequisites. |
| Post Conditions | There must be an existing account if creation or editing of user account is done. There must be no existing account if admin/official deletes account. | |
| Special Requirements |  | |
| Extension Points | Usability of account may be blocked and/or refer to Receivable and Payable use case. | |

|  |  |  |
| --- | --- | --- |
| Set Departmental Equipment and Research Fund | | |
| Description | This is where budget that is set aside for other things get taken into consideration. | |
| Pre-Conditions | User must be an Administrator or Official to access this used case. | |
| Flows | Basic or Normal Flows | 1 Tally fund from university necessities such as equipment.  2 Tally any requested research fund.  3 Admin/Official may view this resulting sum. |
|  | Alternative Flows |  |
| Post Conditions | The sum of fund must reflect on wider university system. | |
| Special Requirements |  | |
| Extension Points |  | |

|  |  |  |
| --- | --- | --- |
| Account Receivable and Payable Management | | |
| Description | This is where students pay their tuition, employees receive their payments, and Officials pay third party for their services. | |
| Pre-Conditions |  | |
| Flows | Basic or Normal Flows | 1 If student, s/he will view his/her balance. If official, view third party’s bill or view all employees pending paychecks. If professor, view paycheck.  2 Student may enter account s/he wishes to pay. If official, enter payment amount for third party transaction or send out employee paychecks.  3 If student, send student notification. If official, log transaction. |
|  | Alternative Flows | 1 Invalid amount: re-prompt student or official to enter amount from step 2 above.  2 Invalid date: Officials may only send paychecks out on certain dates (step 2 above). |
| Post Conditions | System logs and/or notifies via email of any transaction. | |
| Special Requirements |  | |
| Extension Points |  | |

|  |  |  |
| --- | --- | --- |
| Communicate | | |
| Description | Mailing and notifications will be handled here. | |
| Pre-Conditions | Depending on reason for mail or notification, this action might be invoked by another used case. | |
| Flows | Basic or Normal Flows | 1 If any used case perform an action, mail recipient (or the user that that action was performed on).  2 Officials may access emailing directly and send out emails to whoever they wish to send to. |
|  | Alternative Flows | 1 Account does not exist: send an error email back to the sender. |
| Post Conditions | A user gets a mail. | |
| Special Requirements |  | |
| Extension Points | Used cases from Registrar or Financial categories may access this use case at step 1. | |

|  |  |  |
| --- | --- | --- |
| Perform Maintenance | | |
| Description | Administrator will be allowed to shut system down and debug system or perform updates as necessary. | |
| Pre-Conditions | User must be Administrator. | |
| Flows | Basic or Normal Flows | 1 User can debug entire system |
|  | Alternative Flows | 1 Restart system. |
| Post Conditions | System must be up and running. | |
| Special Requirements | System must be shut down or unavailable to other users. | |
| Extension Points |  | |

6.2 USER INTERFACE REQUIREMENTS

The user interface will be aesthetically appealing, simple, fast and flexible, where it can seamlessly interface with any web browser on any PC, MAC, IOS, and Android device. The login page will be as simple as just prompting the user for their credentials, while displaying the university’s logo and colors. The following page will prompt the user with a variety of menu options that directs the user to the appropriate page, while also displaying university logo and colors.

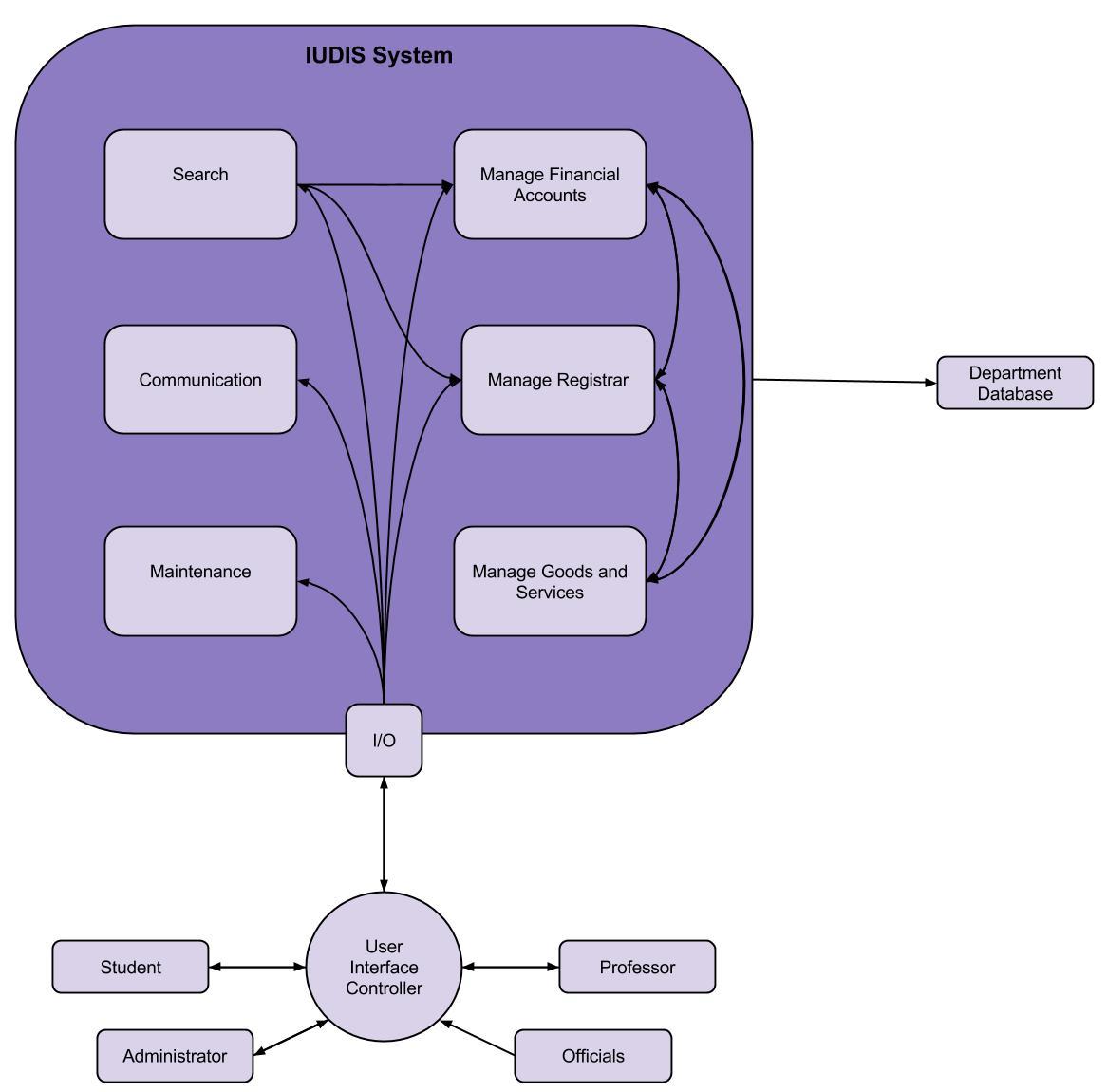
6.3 COMPONENT (COMPONENT/PACKAGE/SUBSYSTEM) ARCHITECTURE

*6.3.1 Component Descriptions*

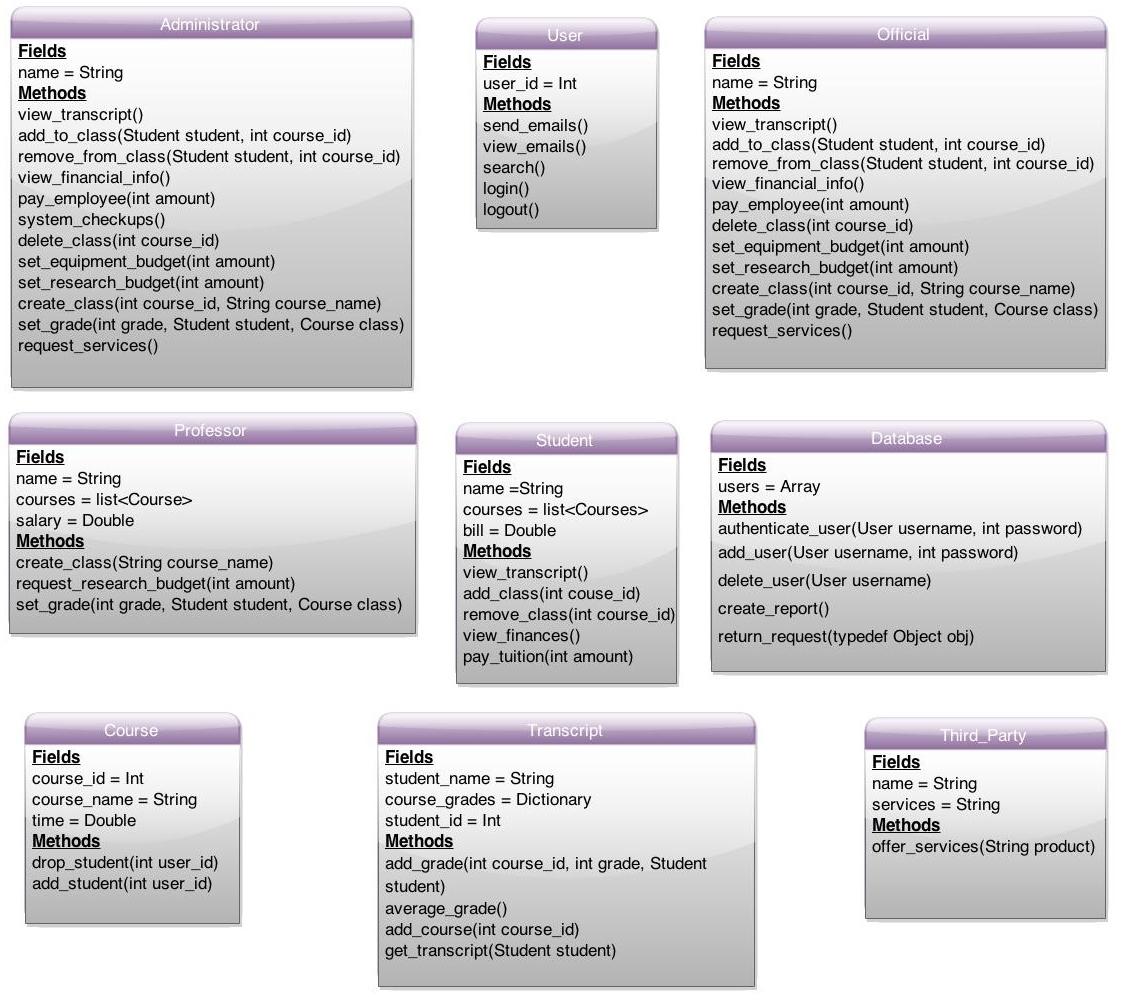
The architecture of the IUDIS is displayed in section 6.3.2, where the users will interact with the User Interface Controller (UIC). The UIC will then prompt access to the users through the set methods of the I/O and IUDIS depending on the level of permission granted by the Administrator.

Simply put the UIC allows access to the IUDIS system, which enables access and updates to the database.

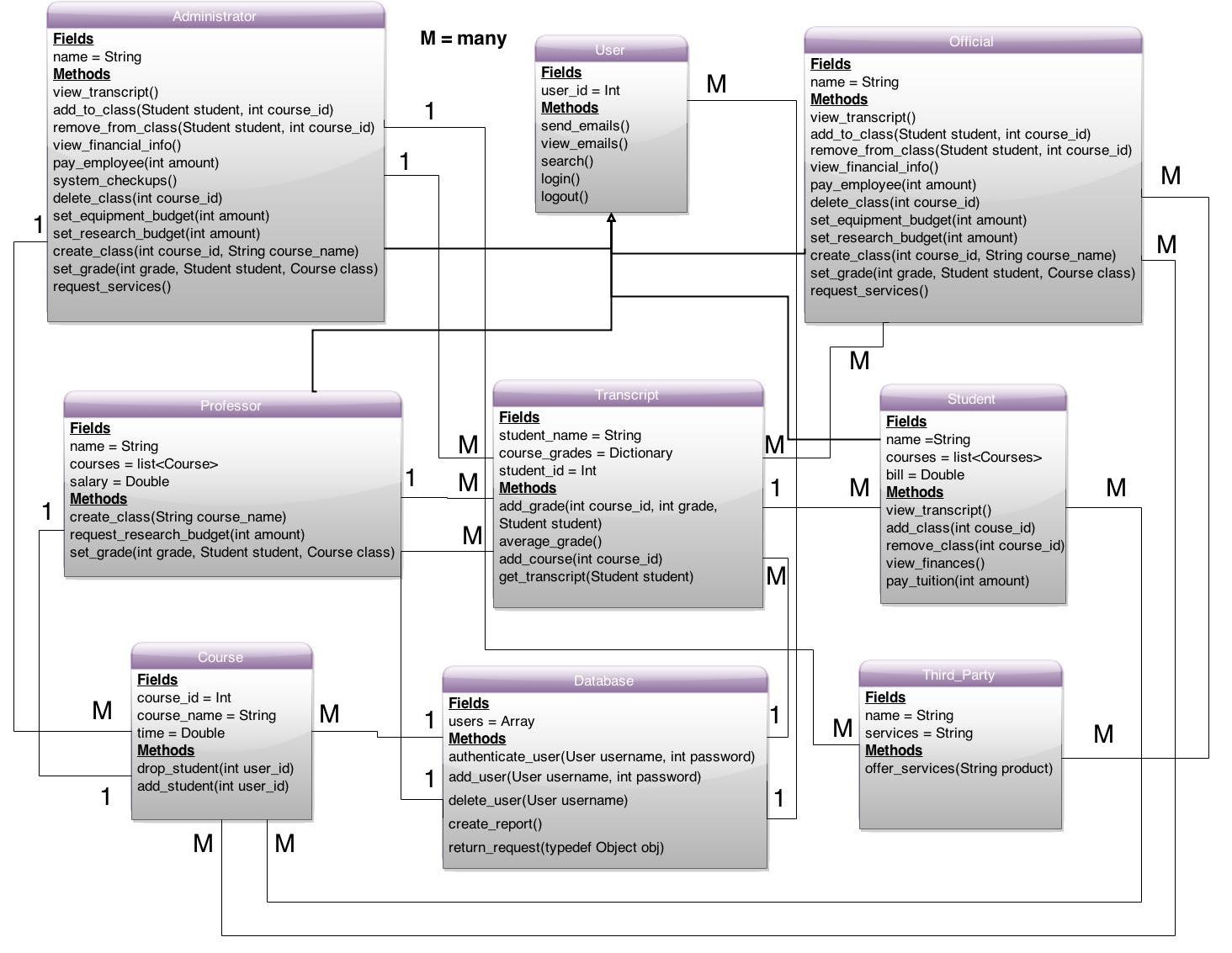
*6.3.2 Component Architecture Diagram*



6.4 CLASS DIAGRAMS



6.5 CLASS RELATIONSHIP/INTERACTION DIAGRAMS



6.6 EVENT SECTION

*6.6.1 Event Dictionary*

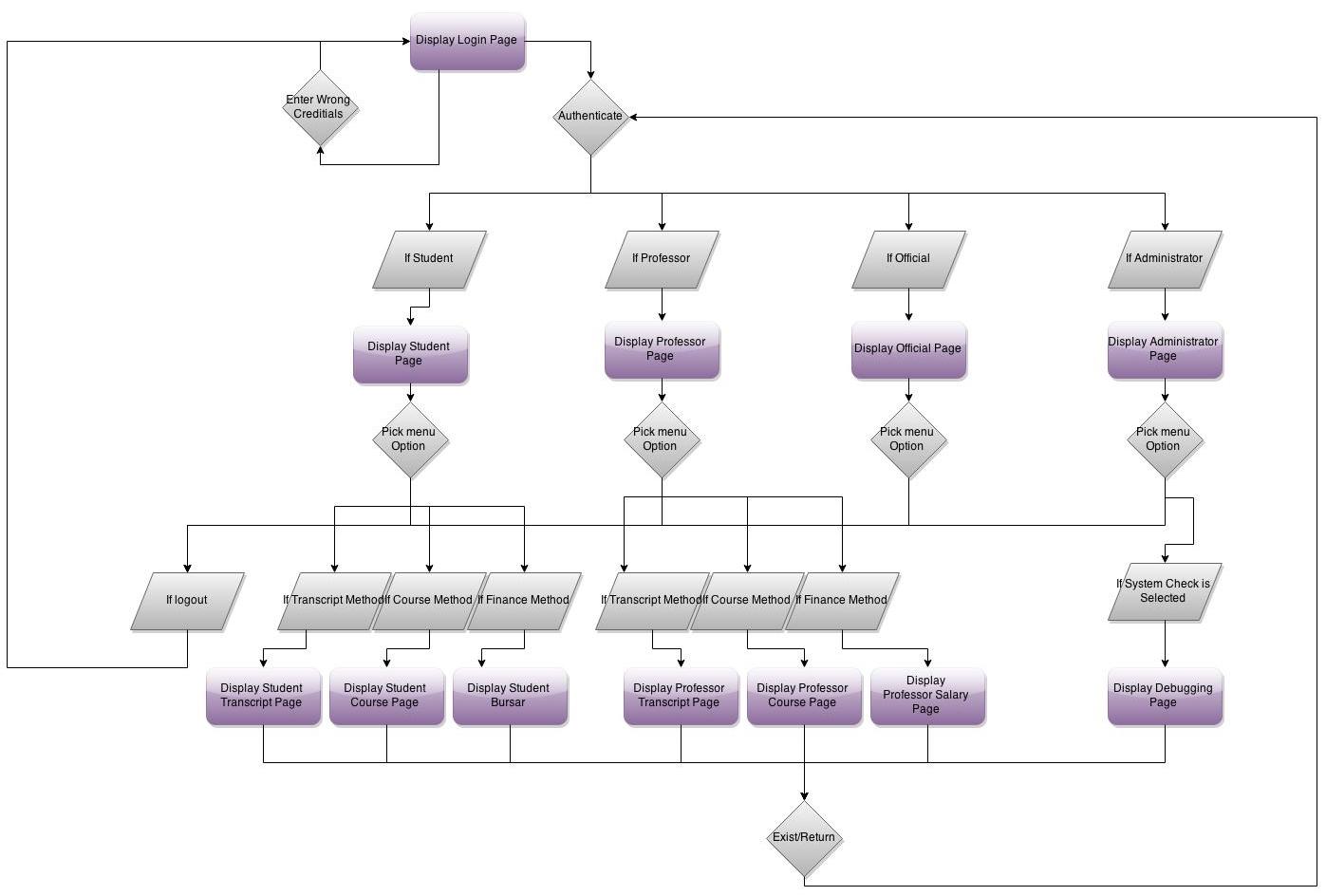
|  |  |  |  |
| --- | --- | --- | --- |
| Authenticate | Determine who the user is depending on their username. From there, a user will see student, professor, official, or administrator page | Login Page | Student Page, Professor Page, Official Page, or Administrator Page |

|  |  |  |  |
| --- | --- | --- | --- |
| Enter Wrong Credentials | Return’s user back to the login page | Login Page | Login Page |

|  |  |  |  |
| --- | --- | --- | --- |
| Pick Menu Option | Displays a variety of options that a user can choose | Login Page | Student Transcript Page, Student Course Page, Student Bursar, Professor Transcript Page, Professor Course Page, Professor Salary Page, Debugging Page |

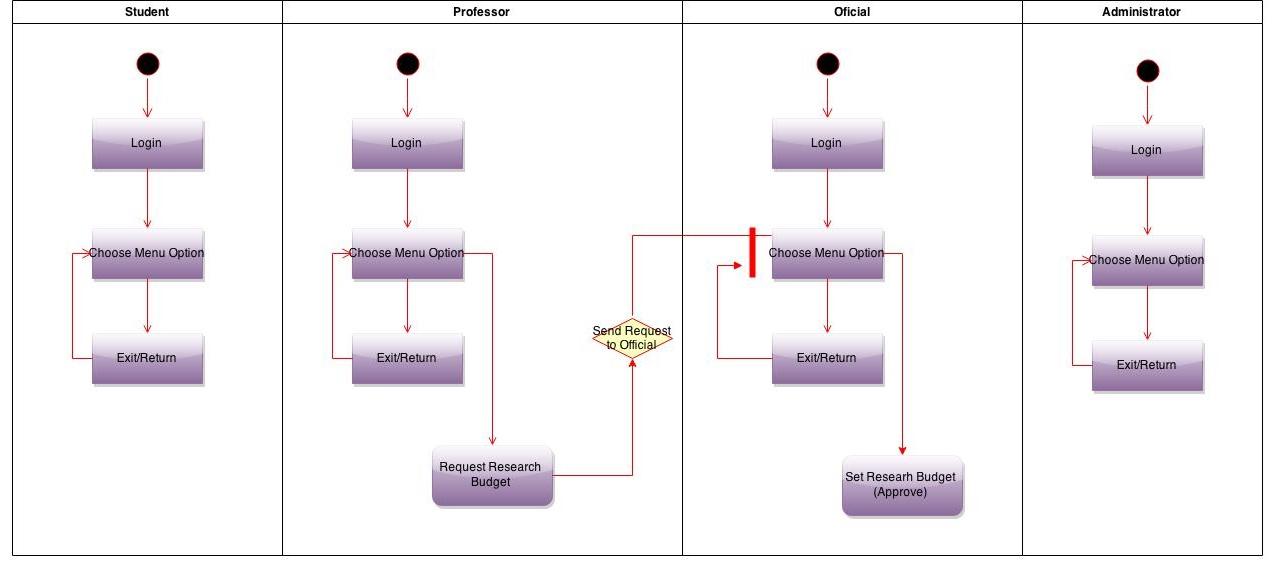
|  |  |  |  |
| --- | --- | --- | --- |
| Exit/Return | After a user is finished with whatever page s/he is currently on, the user will then go back to their original page. | Student Transcript Page, Student Course Page, Student Bursar, Professor Transcript Page, Professor Course Page, Professor Salary Page, Debugging Page | Student Page, Professor Page, Official Page, or Administrator Page |

*6.6.2 Event Diagrams*



6.7 ACTIVITY/STATE (SCENARIO) SECTION

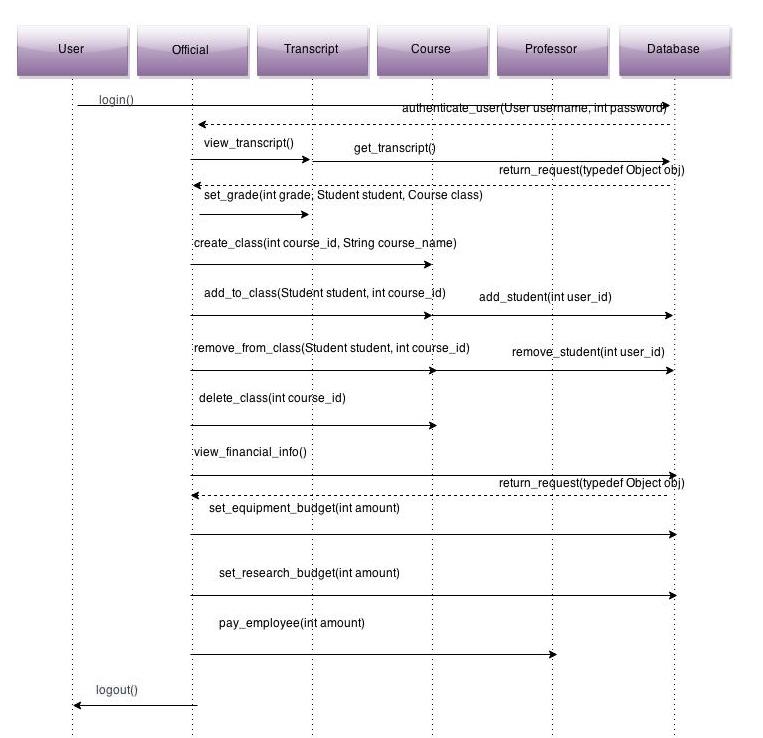
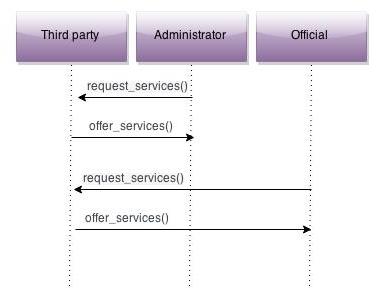
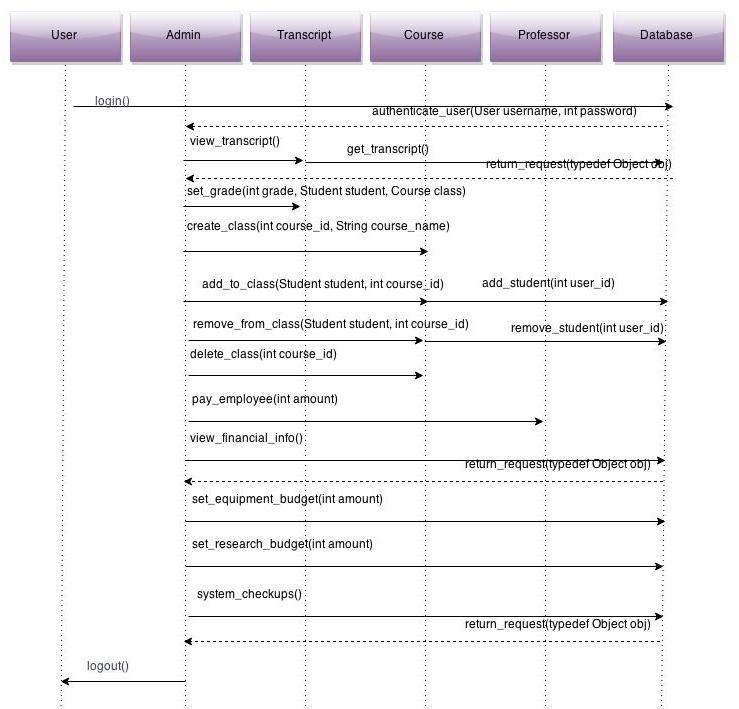
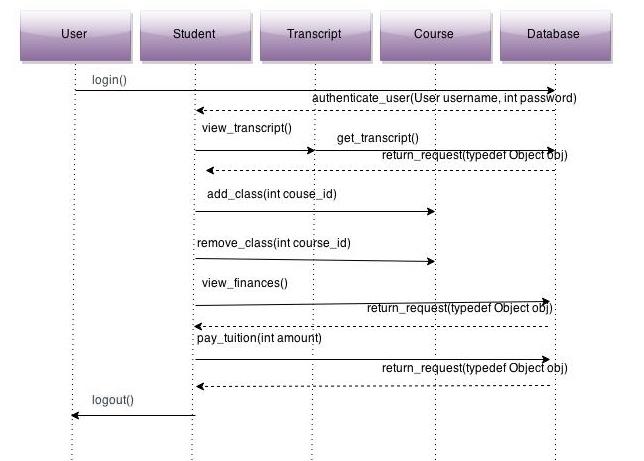
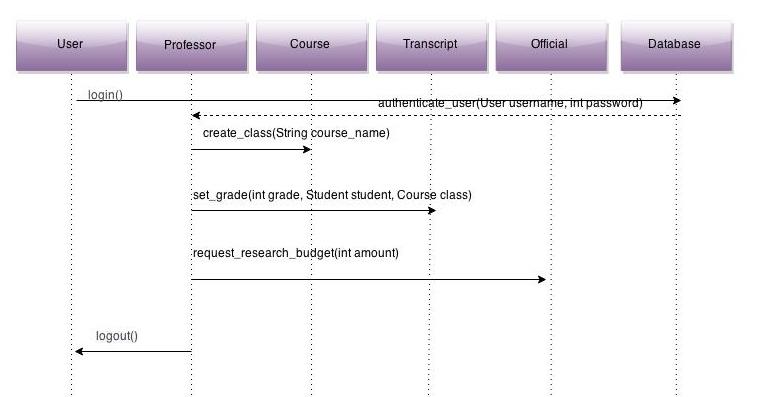
*6.7.1 Activity (Scenario) Diagrams*



*6.7.2 Activity (Scenario) Specification*

The activity diagram demonstrates one of many scenarios where the professor can request a financing option for his research from the university officials. The official will then approve the professor’s research funds. The other activities will be illustrated in the SDD.

6.8 SEQUENCE DIAGRAMS



6.9 COLLABORATION DIAGRAMS

This will be discussed in the SDD.

6.10 DICTIONARIES

Class Dictionaries

|  |  |  |  |
| --- | --- | --- | --- |
| User | Serves as a base for four classes | user\_id = int | send\_emails()  view\_emails()  search()  login()  logout() |

|  |  |  |  |
| --- | --- | --- | --- |
| Student | Gives functionality specific to students | name = String  courses = list<Courses>  bill = double | view\_transcript()  add\_class(int course\_id)  remove\_class(int course\_id)  view\_finances()  pay\_tuiton(int amount) |

|  |  |  |  |
| --- | --- | --- | --- |
| Professor | Gives functionality specific to professors | name = String  courses = list<Courses>  salary = double | create\_class(String class\_name)  request\_research\_budget(int amount)  set\_grade(int grade, Student student, Course class) |

|  |  |  |  |
| --- | --- | --- | --- |
| Official | Gives functionality specific to officials | name = String | view\_transcript()  add\_to\_class(Student student, int course\_id)  remove\_from\_class(Student student, int course\_id)  view\_financial\_info()  pay\_employee(int amount)  delete\_class(int course\_id)  set\_equipment\_budget(int amount)  set\_research\_budget(int amount)  create\_class(int course\_id, String course\_name)  set\_grade(int grade, Student student, Course class)  request\_services() |

|  |  |  |  |
| --- | --- | --- | --- |
| Administrator | Gives functionality specific to administrators | name = String | view\_transcript()  add\_to\_class(Student student, int course\_id)  remove\_from\_class(Student student, int course\_id)  view\_financial\_info()  pay\_employee(int amount)  system\_checkups()  delete\_class(int course\_id)  set\_equipment\_budget(int amount)  set\_research\_budget(int amount)  create\_class(int course\_id, String course\_name)  set\_grade(int grade, Student student, Course class)  request\_services() |

|  |  |  |  |
| --- | --- | --- | --- |
| Third\_Party | Gives functionality specific to third party | name = String  services = String | offer\_services() |

|  |  |  |  |
| --- | --- | --- | --- |
| Transcript | Offers accessing and editing abilities of student transcripts for certain classes | student\_name = String  course\_grades = Dictionary  student\_id = int | add\_grade(int course, int grade, Student student)  average\_grade()  add\_course(int course\_id, Student student)  get\_transcript(Student student) |

|  |  |  |  |
| --- | --- | --- | --- |
| Course | Provides functionality to edit courses and associate courses with classes | course\_id = int  course\_name = String  time = double | add\_student(int user\_id)  drop\_student(int user\_id) |

|  |  |  |  |
| --- | --- | --- | --- |
| Database | Class that hands all storage abilities for other classes | users = Array | authenticate\_user(User username, int password)  add\_user(User username, int password)  delete\_user(User username)  create\_report()  return\_request(typedef Object obj) |

Method Dictionaries

|  |  |  |  |
| --- | --- | --- | --- |
| send\_email | Allows communication between other users | None | User |

|  |  |  |  |
| --- | --- | --- | --- |
| view\_email | Views messages | None | User |

|  |  |  |  |
| --- | --- | --- | --- |
| search | Allows for the exploration through the universities databases | None | User |

|  |  |  |  |
| --- | --- | --- | --- |
| login | Allows access to the university database depending on permissions | None | User |

|  |  |  |  |
| --- | --- | --- | --- |
| view\_transcript | Views grades on transcript | None | Student, Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| add\_class | Allows for the addition of courses | int course\_id | Student |

|  |  |  |  |
| --- | --- | --- | --- |
| remove\_class | Allows for the removal of courses | int course\_id | Student |

|  |  |  |  |
| --- | --- | --- | --- |
| view\_finances | View payables to university | None | Student, |

|  |  |  |  |
| --- | --- | --- | --- |
| pay\_tuition | Allows student to pay tuition | int amount | Student |

|  |  |  |  |
| --- | --- | --- | --- |
| create\_class | Allows professors to create a class | String course\_name | Professor |

|  |  |  |  |
| --- | --- | --- | --- |
| request\_research\_budget | Allows for a professor to request a research budget | int amount | Professor |

|  |  |  |  |
| --- | --- | --- | --- |
| set\_grade | Gives a grade to a student | int grade, Student student, Course class | Professor, Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| add\_to\_class | Adds a student to a course | Student student, int course\_id | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| remove\_from\_class | Removes a student from a course | Student student, int course\_id | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| view\_financial\_info | Allows for the viewing of the whole university’s finances | None | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| pay\_employee | pays employee | int amount | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| system\_checkups | Reports systems failures | None | Administrator |

|  |  |  |  |
| --- | --- | --- | --- |
| delete\_class | Removes a class from the Database | int course\_id | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| set\_equiptment\_budget | Sets the equipment budget for professors that need to buy material for labs and such | int amount | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| set\_research\_budget | Sets the research budget for professors that need to buy material for student’s pay and materials | int amount | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| create\_class | creates a class on the database | int course\_id, String course\_name | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| set\_grade | Sets the grade on the database for the student | int grade, Student student, Course class | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| request\_services | Allows for the request of services from third party entities | None | Administrator, Official |

|  |  |  |  |
| --- | --- | --- | --- |
| offer\_services | Allows third party entities to offer services to the admin or officials within a university | None | Third\_party |

|  |  |  |  |
| --- | --- | --- | --- |
| add\_grade | Adds a grade for a course to a student’s transcript | int course\_id, int grade, Student student | Transcript |

|  |  |  |  |
| --- | --- | --- | --- |
| avrage\_grade | gives the averages of the student’s grades | None | Transcript |

|  |  |  |  |
| --- | --- | --- | --- |
| add\_course | Adds a course to a student's transcript | int course\_id | Transcript |

|  |  |  |  |
| --- | --- | --- | --- |
| get\_transcript | Returns the students transcript from the database | Student student | Transcript |

|  |  |  |  |
| --- | --- | --- | --- |
| drop\_student | Removes student from a course | int user\_id | Courses |

|  |  |  |  |
| --- | --- | --- | --- |
| add\_student | Adds a student to the course | int user\_id | Courses |

|  |  |  |  |
| --- | --- | --- | --- |
| authenticate\_user | Makes sure that the user that is using the system has permissions necessary to access the Database | User username, int Password | Database |

|  |  |  |  |
| --- | --- | --- | --- |
| add\_user | Adds a user to the Database | User username, int Password | Database |

|  |  |  |  |
| --- | --- | --- | --- |
| delete\_user | Removes a user from the database | User username | Database |

|  |  |  |  |
| --- | --- | --- | --- |
| create\_report | Displays system messages | None | Database |

|  |  |  |  |
| --- | --- | --- | --- |
| return\_request | Returns any data off the database to user with correct permissions | typedef Object obj | Database |

Attribute Dictionaries

|  |  |  |  |
| --- | --- | --- | --- |
| user\_id = int | Unique ID to access users | Simple | User |

|  |  |  |  |
| --- | --- | --- | --- |
| name = String | String for user’s name | Simple | Student, Professor, Official, Administrator |

|  |  |  |  |
| --- | --- | --- | --- |
| courses = list<Course> | Stores courses student takes or professor teaches | Simple | Student, Professor |

|  |  |  |  |
| --- | --- | --- | --- |
| bill = double | Student’s tuition amount | Simple | Student |

|  |  |  |  |
| --- | --- | --- | --- |
| salary = double | Professor’s pay | Simple | Professor |

|  |  |  |  |
| --- | --- | --- | --- |
| services = String | Description/Name of product/services | Simple | Third\_Party |

|  |  |  |  |
| --- | --- | --- | --- |
| course\_grades = Dictionary | Holds student’s grades | Simple | Transcript |

|  |  |  |  |
| --- | --- | --- | --- |
| student\_id = int | Unique student ID to get appropriate transcript | Simple | Transcript |

|  |  |  |  |
| --- | --- | --- | --- |
| course\_id = int | This defines a unique class ID | Simple | Course |

|  |  |  |  |
| --- | --- | --- | --- |
| course\_name = String | This defines the class name | Simple | Course |

|  |  |  |  |
| --- | --- | --- | --- |
| time = double | This defines the class time | Simple | Course |

|  |  |  |  |
| --- | --- | --- | --- |
| users = Array | This holds elements or users which this system works on | Simple | Database |

**7. NON-FUNCTIONAL/OPERATIONAL SPECIFICATIONS**

7.1 SYSTEM EXTERNAL INTERFACE REQUIREMENTS

7.2 SAFETY REQUIREMENTS

Anyone who uses the back-end of the system must be trained for operating electrical equipment along with its safety measures.

7.3 SECURITY AND PRIVACY REQUIREMENTS

Database must have authorized persons accessing it. A firewall must be in place to protect external attacks from potential hackers.

7.4 SYSTEM ENVIRONMENT REQUIREMENTS

An indoor environment is required as it is the most suitable environment for electronics and safety.

7.5 COMPUTER RESOURCE REQUIREMENTS

*7.5.1 Computer Hardware Requirements*

The server hardware must be able to support a user base of around 50,000, the average size of a large university.

The minimum server specs are: 4 CPUs 2.6 GHZ and 4GB of RAM and 10TB disk space.

*7.5.2 Computer Hardware Resource Requirements*

The IUDIS should be accessible from any personal or business computer. It is mandatory that each administrator have a computer for this system.

*7.5.3 Computer Software Requirements*

Computer Software Requirements will not be discussed. However, this will be discussed during design.

*7.5.4 Computer Communications Requirements*

Computers and servers will be networked with at least a bandwidth of 50Mb/s.

7.6 SYSTEM QUALITY FACTORS

Reliability, functionality, and usability are most important.

7.7 DESIGN AND CONSTRUCTION CONSTRAINTS

Design and Construction Constraints will not be discussed. However, this will be discussed during design.

7.8 PERSONNEL-RELATED REQUIREMENTS

It is mandatory that personnel are experts at programming. Specific requirements for personnel will be discussed later in development.

7.9 TRAINING-RELATED REQUIREMENTS

The IUDIS must be user-friendly to basic users. Administrators, Officials, Professors, and students are all from different realms of knowledge. Ideally, the IUDIS should be intuitive to use for everyone.

7.10 LOGISTICS-RELATED REQUIREMENTS

Logistics-Related Requirements will not be discussed.

7.11 PACKAGING REQUIREMENTS

Packaging Requirements will not be discussed.

7.12 PRECEDENCE AND CRITICALITY REQUIREMENTS

Precedence and Criticality Requirements will not be discussed.

7.13 OTHER REQUIREMENTS

Other Requirements will not be discussed.

**8. SYSTEM TEST PLAN REQUIREMENTS**

SQA will be responsible for ensuring that these documents fulfill the system’s requirements. To do this, the SQA team will generate test cases for our system and run them through the documentation to see if they can be worked through. SQA team will continue to generate test cases for our system as well as review the documentation to ensure it fulfills the client’s needs. The SQA may require a sample database to run their test cases on all processes. Then, a set non-programmers will test the user interface.

**9. QUALIFICATION PROVISIONS**

Documents will be maintained by using a specific outline provided by the advisor.

Quality is of utmost importance for this system and its reliability will be what separates it from existing systems. The SQA group will handle Quality Control. They will test using plan that they have developed and report to the development team with defects/faults that they have found.

There will be weekly meeting amongst the developers and the clients/client representatives. Work will be verified by tracing back to the SRS. Self-reviews will be done. This is only done by authors of each artifact. Peer reviews are also done similarly. Walkthroughs are done for faults. Inspections are done for detailed reasons and are documented.

**10. REQUIREMENTS TRACEABILITY**

All requirements can be traced to SRS section 7 and project proposal. The requirements for this system have been outlined by the software development team. A document of requirements that have been fulfilled and ones that are currently in progress will be sent with each version release. The interval in which this happens will increase or decrease depending on how on track the system is with what is desired by our buyers. This way the requirement control and its traceability will be documented every step of the way and will be effectively communicated between the developer and the client.

**11. RATIONALE**

The rationale behind the Integrated University Department Information System is to allow a seamless integration between huge university systems and the department system. The computer science department may have many small and independent functions from the main university system and so it should prove to be easy if the department handles its budget by itself while the larger university system tallies the department’s budget into its own budget. Furthermore, students’ grades and registration should be handled likewise.

**12. NOTES**

**13. APPENDICES**

13.1 DICTIONARIES

All Dictionaries are in their respective sections.

13.2 UML DIAGRAMS

All UML Diagrams are in their respective sections.

13.3 SCHEDULE TRACKING

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who | Estimated | Actual | Difference |
| RAS | Munieshwar Ramdass | 10 | 13 | 2 |
| RAS | Ajay Shenoy | 12 | 13 | 1 |
| RAS | August Tan | 7 | 13 | 5 |
|  | Team Summary | 29 | 39 | 10 |

Cumulative

|  |  |  |  |
| --- | --- | --- | --- |
| Who | Estimate | Actual | Difference |
| Munieshwar Ramdass | 10 | 13 | 2 |
| Ajay Shenoy | 12 | 13 | 1 |
| August Tan | 7 | 13 | 5 |
| Team Summary | 29 | 13 | 10 |

13.4 DEFECT TRACKING

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who | Estimated | Actual | Difference |
| RAS | Munieshwar Ramdass | 5 |  |  |
| RAS | Ajay Shenoy | 4 |  |  |
| RAS | August Tan | 6 |  |  |
|  | Team Summary | 15 |  |  |

Cumulative

|  |  |  |  |
| --- | --- | --- | --- |
| Who | Estimate | Actual | Difference |
| Munieshwar Ramdass | 10 |  |  |
| Ajay Shenoy | 12 |  |  |
| August Tan | 7 |  |  |
| Team Summary | 29 |  |  |

Gantt Chart

