OMNICAL FINAL DOCUMENTATION

J MULLER	26058995
TB BENSCH	26175940
R HAMMOND	26240343
MC ERASMUS	26438585
TR VILLET	26118653

Table of Contents

PROJECT DESCRIPTION

OmniCal is a powerful piece of day-planning, calendar-style academic organizer software, designed with students and lecturers.

OmniCal Allows the user to organize and put their time to better use, through arranging events and even organizing their timetables, with the help of this easy-to-use and easy-to-understand software. OmniCal can automatically search and develop a timetable for a student or lecturer through means of academic records, extra modules as well as manual entries (for example, SI or assistant classes):

- A login system will be used. The user will be able to log on and off when desired.
- A user-friendly, calendar-styled day planner will display a daily schedule.
- It will allow quick and easy access to all your planned events and flexible editing of all events.
- It will have an intuitive, easy-to-understand user interface.
- It will allow personal and academic use, not limiting users to one or the other.
- It is simple and light on hardware, not requiring much processing power.
- It will allow students and lecturers to be up to date with all their planned activities, classes, tests, due-dates, appointments and other events.
- A timetable will be created on the timetable screen once the user has logged in for the first time. The program will load the desired timetable from a database and the user can add any extra classes to their timetable.
- The user will be able to edit their timetable whenever they desire to.
- OmniCal will also keep track of extra-curricular activities as well as provide suitable diets.

SYSTEM REQUIREMENTS

HARDWARE:

- A computer with relevant input devices
- Optimal RAM space. OmniCal does not require immense RAM space but for a smooth running system, it is advised to have capable free RAM space.
- Internet connection for any updates that may be required.

SOFTWARE:

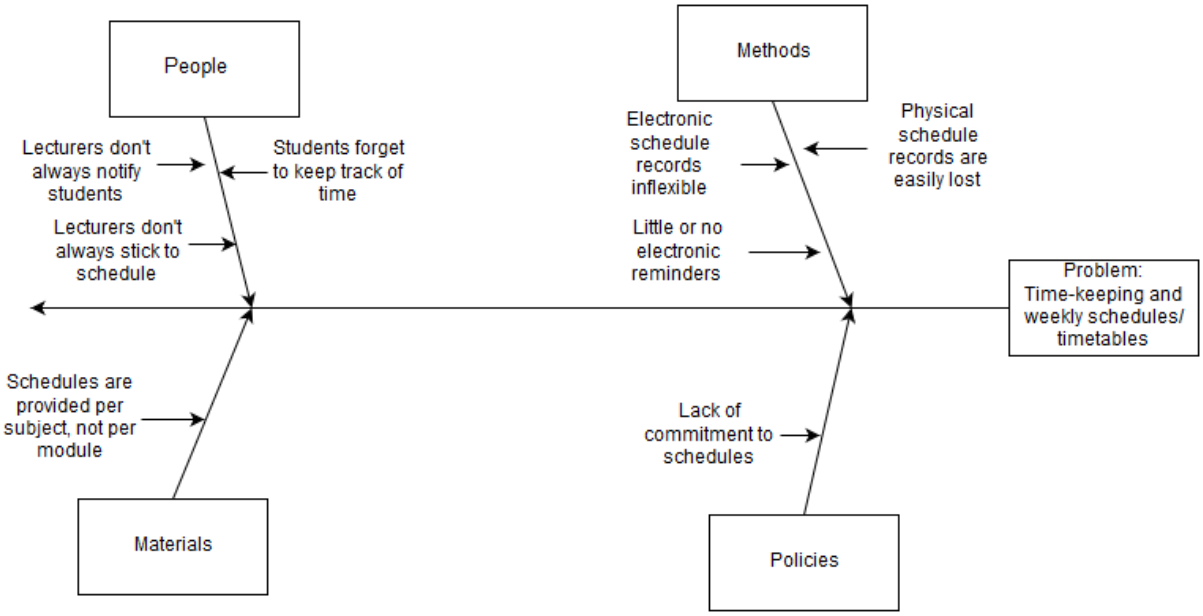
- Operating systems like Windows, Apple and Linux are supported.
- The relevant software would need to be installed in order for the program to be executed, for example, Notepad and CaptureScreen Lite.
- For online storage of the schedule, cloud software or something equivalent would have to be installed.

OTHER:

- A basic understanding of computers and computer interface is required.
- The user should be literate.
- If the user is a student, they will only be accommodated if they meet the following requirements:
    - A registered student at North West University Vaal Triangle Campus
    - Currently studying BScInformationTechnology
    - Current Academic year first, second or third year
- If the user is an administrator, the appropriate password and further system information will be required.
- A username and password will be required for any user who uses this system.
- Relevant files cannot be removed and are therefore, required for the program to function correctly, for example, DO NOT DELETE ANY DATABASES.
- The user has to be either English, Afrikaans, Zulu or Sesotho to understand and be able to use the program.

By: TR Villet & J Muller

ISHIKAWA (FISHBONE) DIAGRAM



By: MC Erasmus

PIECES FRAMEWORK

- **Performance** - Does the system provide an easy and effective result for the users?
  The system provides an improved day to day planning without struggle.

- **Information** - Does the system private useful and accurate information?
  The system provide a better validation of the user's data and is presented in a well formatted display.

- **Economics** – Is the system cost-effective?
  We made sure that the time and money is spend as effectively as possible to ensure that the system is cost effective.

- **Control –** Does the system provide the user with privacy and security?
  The system uses a Staff or Student number as identification and a password to gain access to the systems functions.

- **Efficiency –** Is the Flow of events and data storage implemented so that it is fully optimized?
  The flow of event id designed so that the user can access and uses the system in a fast and effected manner. The database is designed to reduce redundancy for safe data storage and transfer.

- **Service –** Does the system provide a useful service?
  The system provides the user with a quick and easy way to plan out his or her day.
  The system can also be used in a option of multiple languages if the user wants the system in their preferred language.

PRIORITY RANKING MATRIX

OMNICAL PRIORITY RANKING MATRIX									
USE CASE NAMES	Ranking Criteria (1 - 5)						Total	Priority	Build Cycle
	1	2	3	4	5	6			
Login	5	5	2	1	1	1	15	Medium	3
Sign up	5	4	2	1	1	1	14	Medium	3
Log out	3	3	2	1	1	1	11	Low	4
Get database info From database	2	5	5	3	1	1	17	High	2
Send info to database	2	5	4	3	1	1	16	High	1
Create event	4	4	5	2	1	1	17	High	2
View in day plann	5	5	4	2	1	1	18	High	1

By: Everyone in the group

PROBLEM STATEMENT MATRIX

PROBLEM STATEMENT MATRIX

PROJECT: OmniCal	PROJECT MANAGER: Jacqueline Muller
CREATED BY: J Muller 26058995	LAST UPDATED BY: J Muller
DATE CREATED: 02/26/2016	DATE LAST UPDATED: MM/DD/YYYY

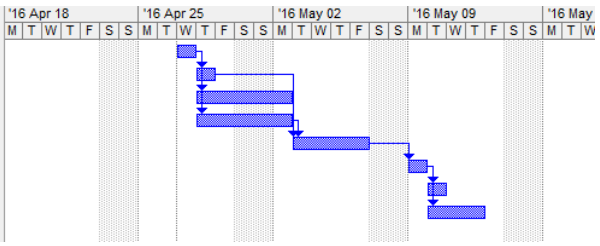
Brief Statements of Problem, Opportunity, or Directive	Urgency	Visibility	Annual Benefits	Priority or Rank	Proposed Solution
A graphical user interface must be designed.	2 Weeks	High	N/A	1	Research GUI options
Coding must be done using one of two IDEs	3 Weeks	Low	N/A	2	Work together to find a suitable IDE
Every possible error or programming exception must be handled.	2 Weeks	Medium	N/A	3	Research different examples of exception handling/ Trial and error
A database may have to be implemented for storing and accessing multiple users' information.	2 Weeks	Medium	N/A	1	Design a database suitable for storing all relevant information
User-friendliness is a subjective preference. Testers may be required to provide feedback.	1 Weeks	High	N/A	4	Trial and error / Set up a survey which will receive feedback from users.

By: J Muller

GANTT CHART

ACTIVITY	DAYS	PRECEDORS
1 = Planning	1	NONE
2 = Documentation	3	1
3 = Design GUI	3	1
4 = Design Database	1	1
5 = Code Functionality	4	3,4
6 = Testing	1	5
7 = Feedback	1	6
8 = Improvements	3	6

	Task Name	Duration	Start	Finish	Predecessors
1	Planning	1 day	Wed 16-04-27	Wed 16-04-27	
4	Design Database	1 day	Thu 16-04-28	Thu 16-04-28	1
2	Documentation	3 days	Thu 16-04-28	Mon 16-05-02	1
3	Design GUI	3 days	Thu 16-04-28	Mon 16-05-02	1
5	Coding	4 days	Tue 16-05-03	Fri 16-05-06	3;4
6	Testing	1 day	Mon 16-05-09	Mon 16-05-09	5
7	Feedback	1 day	Tue 16-05-10	Tue 16-05-10	6
8	Improvements	3 days?	Tue 16-05-10	Thu 16-05-12	6



By: MC Erasmus

Paths:

- 1-2
- 1-3-5-6-7
- **1-3-5-6-8**
- 1-4-3-5-6-7
- 1-4-3-5-6-8

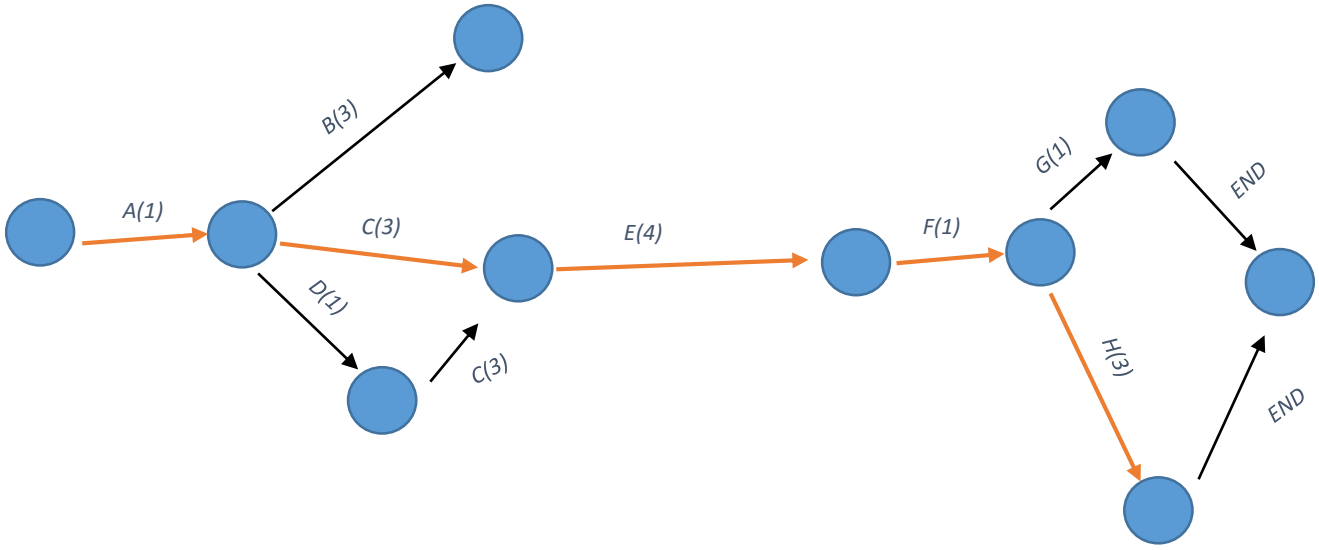
Critical path:

- 1-3-5-6-8

ACTIVITY ON ARROW

ACTIVITY	DAYS	PRECEDORS
A = Planning	1	NONE
B = Documentation	3	A
C = Design GUI	3	A
D = Design Database	1	A
E = Code Functionality	4	C,D
F = Testing	1	E
G = Feedback	1	F
H = Improvements	3	F

ACTIVITY-ON-ARROW



By: R Du Plessis

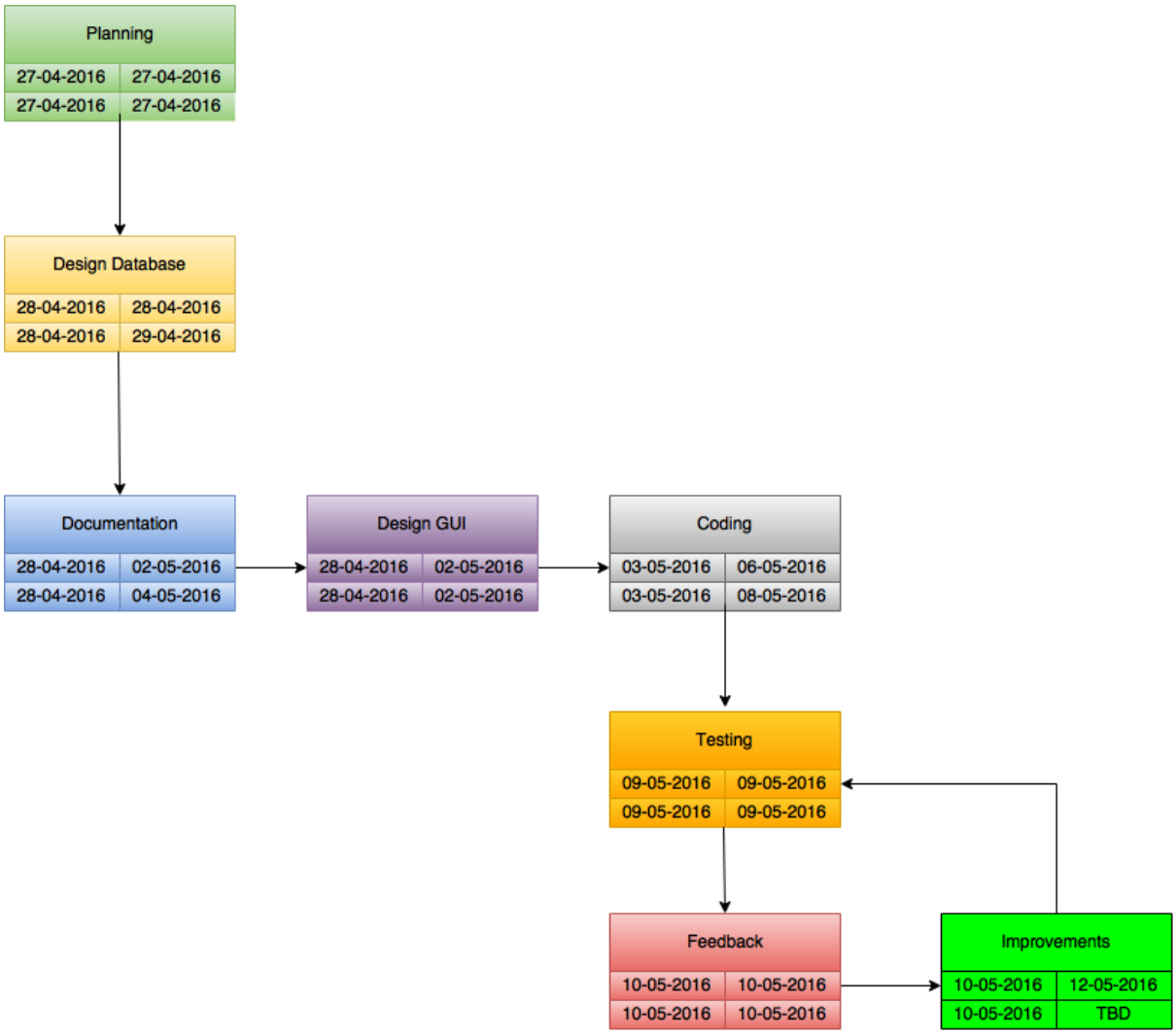
Paths:

- A-B
- A-C-E-F-G
- **A-C-E-F-H**
- A-D-C-E-F-G
- A-D-C-E-F-H

Critical path:

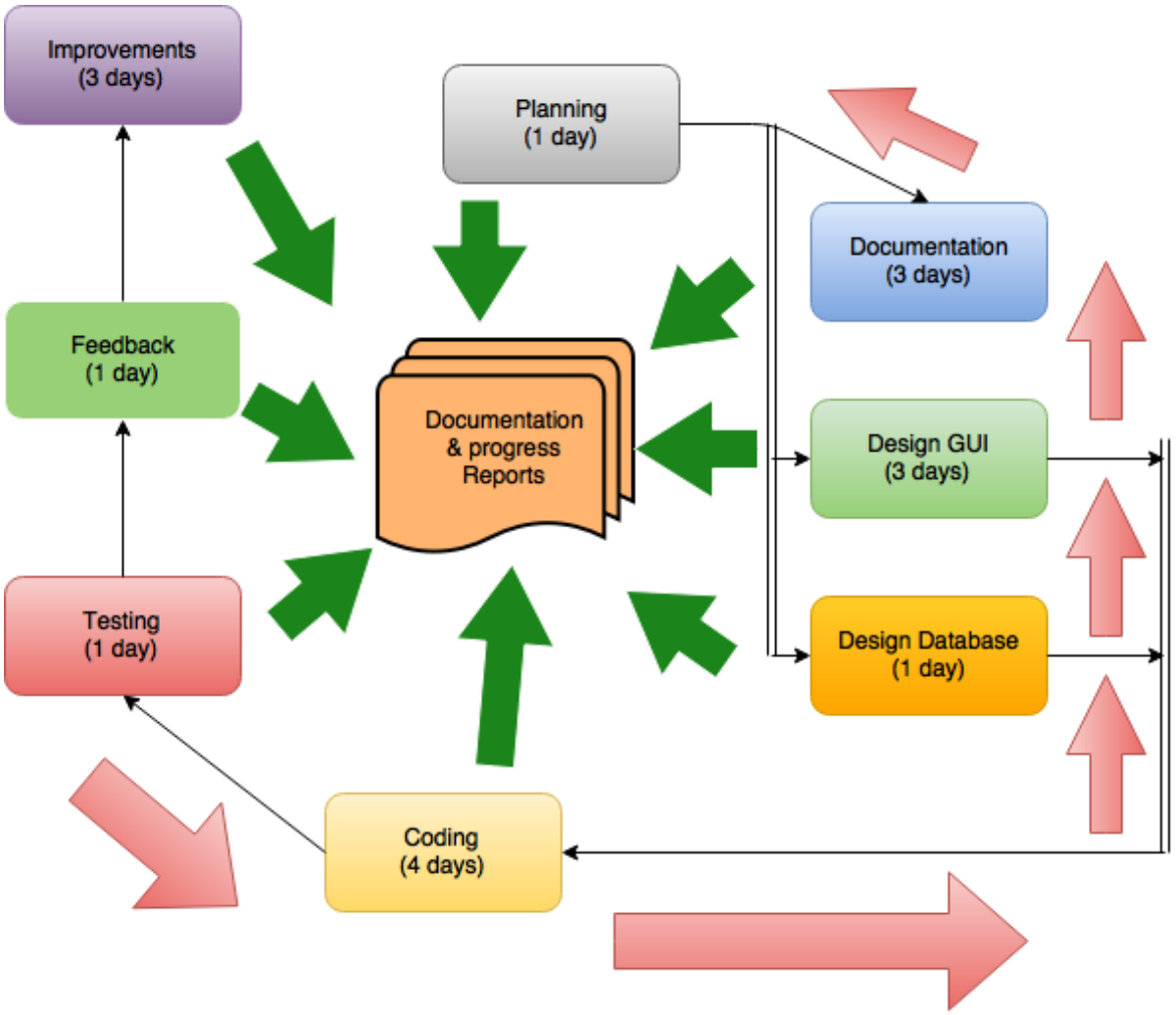
- A-C-E-F-H

PERT CHART



By: R Du Plooy

SYSTEM DEVELOPMENT METHODOLOGIES



By: TB Bensch

EXPECTATIONS MANAGEMENT MATRIX

Priorities → / ↓Measures of success	Max or min	Constrain	Accept
Cost			√
Schedule	√		
Scope and/or Quality		√	

By: TB Bensch

LECTUER EXPECTATIONS QUESTIONNAIRES (SAMPLE)

1. Do you like the idea of a day to day academic planner for students, lecturers and workers of the NWU Vaal Triangle campus?

Yes	No

2. Do you think the system will help students better perform by planning there events on the system?

Yes	No

3. Should the system be available in other languages?

Yes	No

4. Would you use a day to day planner to plan your day out on campus?

Yes	No

5. If you are an IT student do you want a pre written schedule of your subjects?

Yes	No

6. Should the system include lectures and workers as well?

Yes	No

7. Should the lectures be able to edit the student's timetable to schedule events that is important?

Yes	No

8. Do you think a help function is needed in the system to help the user?

Yes	No

9. Is Omnical a fitting name for the system?

Yes	No

10. Should this system merge with efundi when the system is complete?

Yes	No

BSCIT LECTURER EXPECTATIONS QUESTIONNAIRES RESPONSE

1. Do you like the idea of a day to day academic planner for students, lecturers and workers of the NWU Vaal Triangle campus?

Yes	No

2. Do you think the system will help students better perform by planning there events on the system?

Yes	No

3. Should the system be available in other languages?

Yes	No

4. Would you use a day to day planner to plan your day out on campus?

Yes	No

5. If you are an IT student do you want a pre written schedule of your subjects?

Yes	No

6. Should the system include lectures and workers as well?

Yes	No

7. Should the lectures be able to edit the student's timetable to schedule events that is important?

Yes	No

8. Do you think a help function is needed in the system to help the user?

Yes	No

9. Is Omnical a fitting name for the system?

Yes	No

10. Should this system merge with efundi when the system is complete?

Yes	No

BA LAW LECTURER EXPECTATIONS QUESTIONNAIRES RESPONSE

1. Do you like the idea of a day to day academic planner for students, lecturers and workers of the NWU Vaal Triangle campus?

Yes	No

2. Do you think the system will help students better perform by planning there events on the system?

Yes	No

3. Should the system be available in other languages?

Yes	No

4. Would you use a day to day planner to plan your day out on campus?

Yes	No

5. If you are an IT student do you want a pre written schedule of your subjects?

Yes	No

6. Should the system include lectures and workers as well?

Yes	No

7. Should the lectures be able to edit the student's timetable to schedule events that is important?

Yes	No

8. Do you think a help function is needed in the system to help the user?

Yes	No

9. Is Omnical a fitting name for the system?

Yes	No

10. Should this system merge with efundi when the system is complete?

Yes	No

USER INTERVIEWS

User expectations before prototype (USER 1)

Interviewee: Date: Time: Place: Subject:	Kholofelo Poppy Mocheke 25212192 March 18, 2016 11:00 AM Building 9A-107 Expectations of OmniCal	
Time Allocated	Interviewer Questions or objective	Interviewee Response
2 min.	**Objective** Open interview -Introductions -State purpose of interview	Kholofelo- Responds with greetings and agrees upon the purpose of the interview.
2 min.	**Question 1** Would you make use of a scheduling system daily? **Follow up**	Kholofelo- Yes I would. Planning is an important part of second year studies
2 min.	**Question 2** What do you expect from OmniCal? **Follow up**	Kholofelo- I expect OmniCal to facilitate me and my schedule without hassles and user-unfriendliness
2 min.	**Question 3** What feature would you suggest for OmniCal to progress as a system?	Kholofelo- A cool feature would be being able to schedule my life in my mother tongue especially since I won't always be using OmniCal for school purposes.
8 min.	Time allotted for questions and objectives	
15 min.	Time allotted for interview	

By: J Muller

User prototype experience (USER 1)

Interviewee: Date: Time: Place: Subject:	Kholofelo Poppy Mocheke 25212192 April 20, 2016 1:00 PM Student center Views and opinions of project idea	
Time Allocated	Interviewer Questions or objective	Interviewee Response
2 min.	**Objective** Open interview -Introductions -State purpose of interview	Kholofelo- Responds with greetings and agrees upon the purpose of the interview.
2 min.	**Question 1** Who are the type of people to use our program? **Follow up**	Kholofelo- People with busy lives. People who need to not only plan their lives but order them. People such as Students and the general working public.
2 min.	**Question 2** After looking at the prototype of the project what suggestions could you give us? **Follow up**	Kholofelo- I see that you refer to "Workers" in the program. I feel this is a bit more specific and that the program excludes people who aren't working. Changing this to General would work a lot better.
2 min.	**Question 3** Do you feel that this type of program could be a big help to you as a student? **Follow up**	Kholofelo- Yes, I feel that it would take a big weight off of my shoulders. It would help with the planning of my SI classes and with extra activities that I plan.
3 min.	**Question 4** If you were to have a timetable/schedule saved so that you could have ease of access to it on say your phone, what format would be easiest? **Follow up**	Kholofelo- Personally I wouldn't mind but a lot of people would struggle opening certain types of formats on a mobile device. If we stick to mobile devices exclusively I would say that pdf would be best.
11 min.	Time allotted for questions and objectives	
15 min.	Time allotted for interview	

By: TB Bensch

User expectations before prototype (USER 2)

Interviewee: Date: Time: Place: Subject:	Lizelle De Bruin March 19, 2016 14:00 PM Alberton Spur Expectations of OmniCal	
Time Allocated	Interviewer Questions or objective	Interviewee Response
2 min.	**Objective** Open interview -Introductions -State purpose of interview	Lizelle- Responds with greetings and agrees upon the purpose of the interview.
2 min.	**Question 1** Would you make use of a scheduling system daily? **Follow up**	Lizelle- Yes.
2 min.	**Question 2** What do you expect from OmniCal? **Follow up**	Lizelle- This system must be able to just give me a timetable without irritating me.
2 min.	**Question 3** What feature would you suggest for OmniCal to progress as a system?	Lizelle- I don't know. Giving me a timetable that I can always keep with me. Maybe in a document.
8 min.	Time allotted for questions and objectives	
15 min.	Time allotted for interview	

By: J Muller

User prototype experience (USER 2)

Interviewee: Date: Time: Place: Subject:	Lizelle De Bruin April 19, 2016 3:00 PM Mugg and Bean Expectations of OmniCal	
Time Allocated	Interviewer Questions or objective	Interviewee Response
2 min.	**Objective** Open interview -Introductions -State purpose of interview	Lizelle- Responds with greetings and agrees upon the purpose of the interview.
2 min.	**Question 1** Who are the type of people to use our program? **Follow up**	Lizelle- People who need a schedule obviously
2 min.	**Question 2** After looking at the prototype of the project what suggestions could you give us? **Follow up**	Lizelle- I like it like that. It's simple enough.
2 min.	**Question 3** Do you feel that this type of program could be a big help to you as a general user who works? **Follow up**	Lizelle- Yes.
3 min.	**Question 4** If you were to have a timetable/schedule saved so that you could have ease of access to it on say your phone, what format would be easiest? **Follow up**	Lizelle- A document
11 min.	Time allotted for questions and objectives	
15 min.	Time allotted for interview	

By: TB Bensch

Observations from User interviews

Strategically, the system analyst and project manager chose two users to evaluate OmniCal at two stages of development.

The first interviewee, Kholofelo Poppy Mochele, is a second year BScIT student who is familiar with the development environment. She was chosen because her experience with the development process could give an accurate insight from both a user's and a system's analyst's perspective.

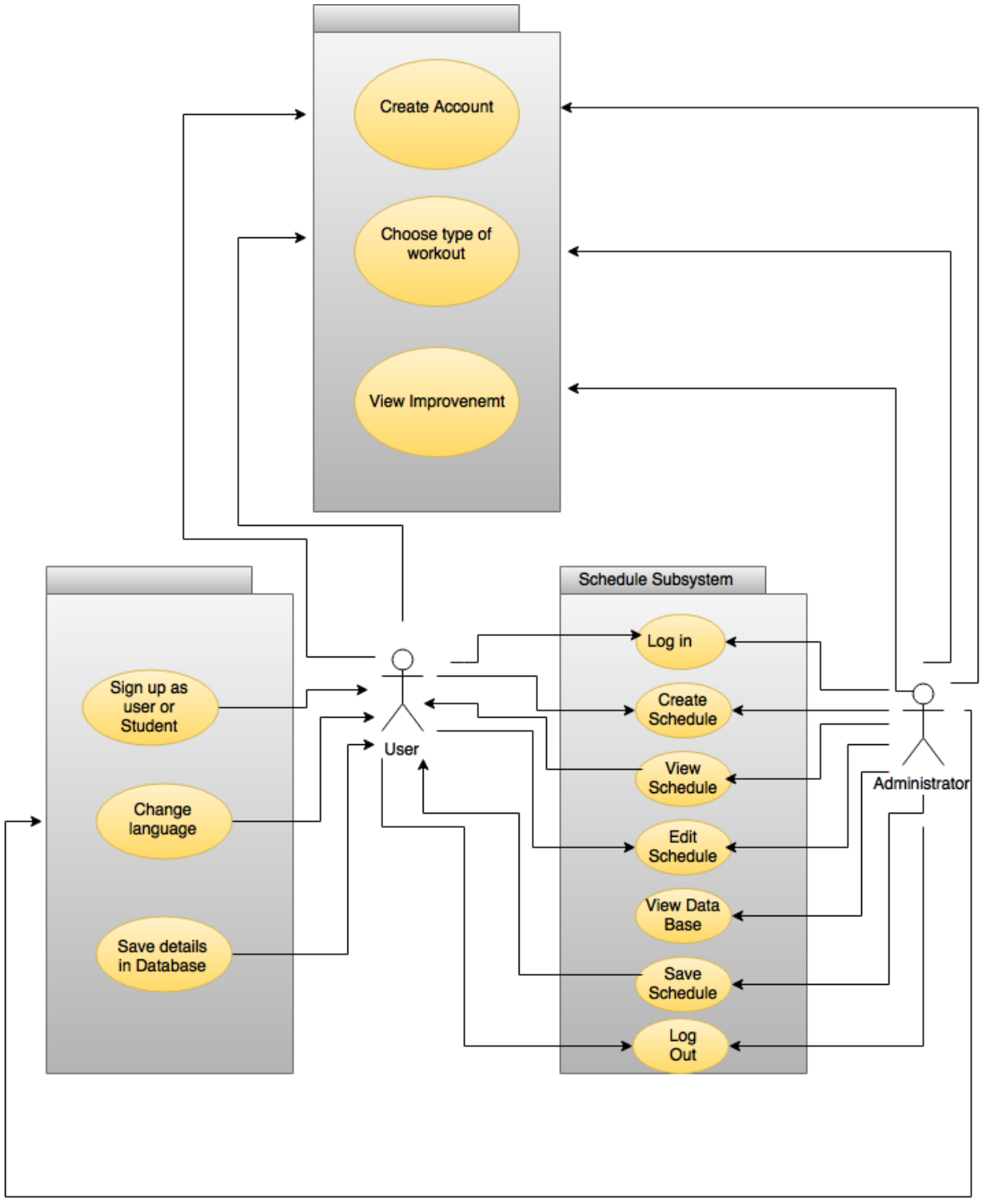
The second interviewee, Lizelle De Bruin, who is not very familiar with IT perspectives. She was chosen due to her lack of knowledge about system development which would give a more accurate indication of general user feedback.

The first interview was conducted by the project manager, J Muller, before the prototyping of the system was underway. The second interview was carried out by the system analyst, TB Bensch, after the user was presented with the prototype to test.

After the four interviews were conducted, the answers from the two different users were compared to provide the best user experience for all users, those who are familiar with the environment and those who are not. It was noted that users do not want a complicated system but rather something that does and effective job in sufficient time with little struggle involved. It was also suggested that having various language options would enhance a suitable user experience. Another suggestion was that the user's schedule be saved in a document, preferably in a PDF format. A useful suggestion was that OmniCal refer to the lecturers as a "General User" rather than "worker" to avoid any confusion.

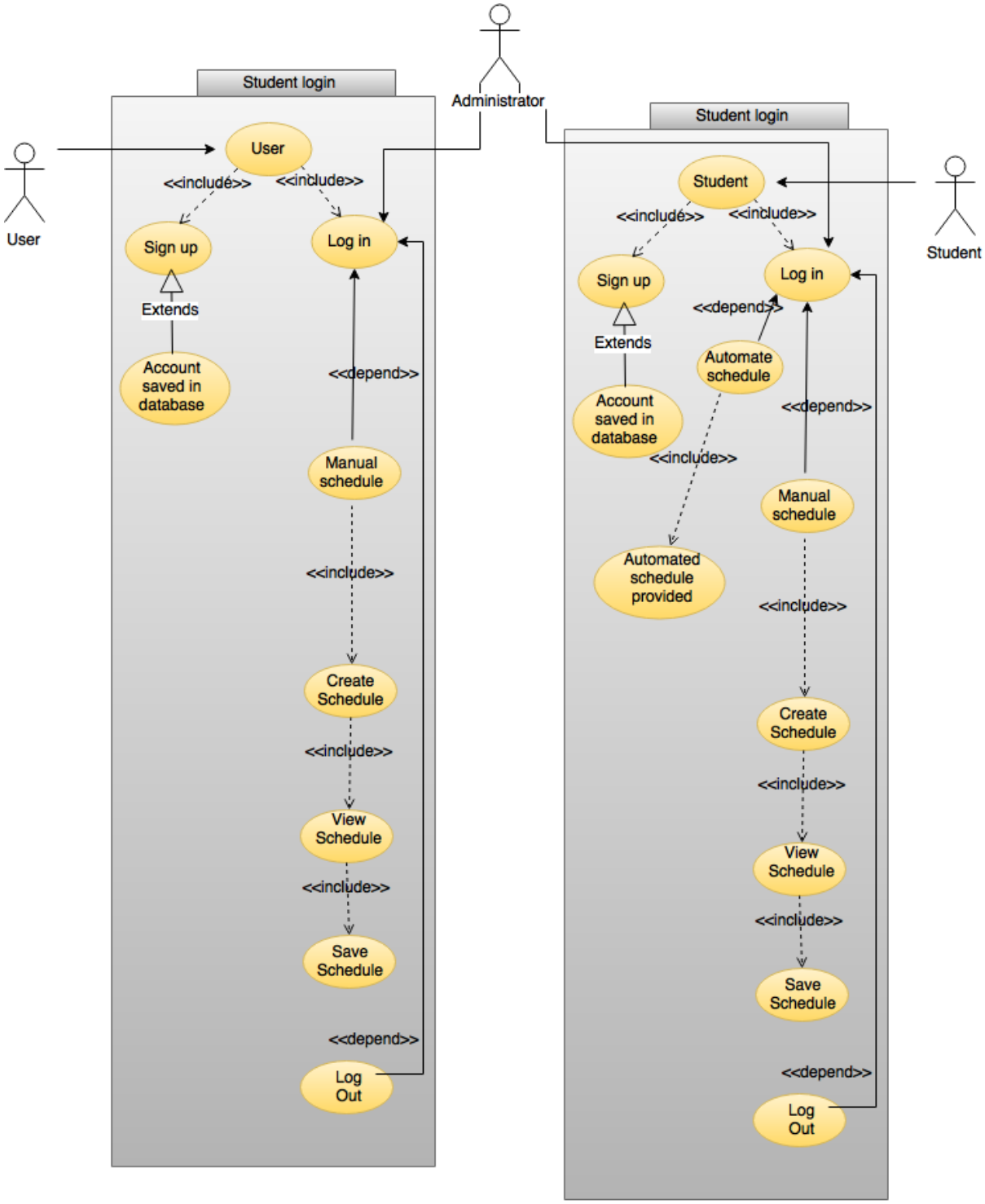
Both interviews turned out fruitful with a few very good suggestions made and views shared from samples of very different user experiences.

USE CASE DIAGRAM



By: R Hammond

USE CASE INCLUDE, EXTEND AND DEPENDS ON DIAGRAM



By: R Hammond

USE CASE NARRATIVES

Receive Information From Database Use Case Narrative

User-Case Name:	Receive information from the database.	User-Case Type Business requirements : Yes
User-Case ID	LO1	
Priority	High	
Source:	Requirement	
Primary Business Actor:	People interested in planning their daily lives.	
Other Participating actors:	-Students -Working class -Sportsmen	
Other interested stakeholders:	-Designer -Programmer: Create the event	
Description	This use-case describes the stakeholders' use of this function in order to receive information from the database to build or edit the current users table.	

By: TB Bensch

Send Information To Database Use Case Narrative

User-Case Name:	Send information from the database.	User-Case Type Business requirements : Yes
User-Case ID	SI01	
Priority	High	
Source:	Requirement	
Primary Business Actor:	Anyone in need of a personal organizer	
Other Participating actors:	- Students - Working class - Sportsmen	
Other interested stakeholders:	-Designer : View Day Planner -Programmer: Create the event	
Description	This use-case describes the event of the designer and the programmer making use of this function in order to send information to the database for storage.	

By: J Muller

Log In Use Case Narrative

Use-Case Name:	Log in	
Use-Case ID:	LG1	
Source:	Graphic user interface	
Primary Business Actor	Registered User	
Description:	The use-case send a password and student number to the database of OmniCal that then will confirm that the user is indeed a client and is registered.	
Precondition:	Only registered users and the main user will gain access to the program to use it properly	
Trigger:	The login use-case will initiate if the login option is selected.	
Typical Courses of Events:	**Actor Action** **Step 1:** The user must create an account. **Step 2 :** The user logs in	**System Response** **Step 3:** The system then checks for valid password and user name **Step 4 :** Login use-case initiates the view day use-case.
Alternate Courses:	n/a	
Conclusion:	The login initiates the database and the view state use-case.	
Post condition:	The Login sends data to the database and receives data from the database and then initiates the view day use-case.	
Business rules:	n/a	
Implementations Constraints and Specifications:	To check the valid passwords and retrieves the correct data for the specific user.	
Assumptions:	The user can only log in if there is an account on that specific	
Open issues:	Posable problems for login with fulltime students and part time students.	

By: R Hammond

Log Out Use Case Narrative

Use Case Name:	Log out	
Use Case ID:	UCID002	
Priority:	Medium	
Source:	Requirement	
Primary Business Actor:	Customer / Client	
Other Participating Actors:	Program itself, logs out and exits	
Other Interested Stakeholders:	System Owners	
Description:	This use case describes the event of a person signing or logging out causing the program to exit.	
Precondition:	The Client or customer must be successfully logged in before they may log out.	
Trigger:	This use case is initiated when a user is finished with the program for the time being, and then clicks "Log out"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user clicks the "log out" button.	**Step 2:** The system responds by displaying the message "Are you sure you wish to log out?" where users can respond in the positive, resulting in a successful log out.
Alternate Course:	**Alt-step 2:** The user responds in the negative, resulting in a cancelation of the log out, thus remaining logged in.	
Conclusion:	This use case concludes when the user either: 1) Completes the log out process Or: 2) Cancels the log out process.	
Post-condition:	None, the user is logged out, and the program is ended	
Business Rules:	➢ When the user is logged out, it must be impossible for anyone not knowing the username or password to the user's account, to gain access to it.	
Implementation Constraints and Specifications:	• GUI to be provided for users must be easy to navigate	
Assumptions:	Database will be easily accessible, and functional.	
Open Issues:	None.	

By: TR Villet

Sign Up Use Case Narrative

Use Case Name:	Sign up.	
Use Case ID:	SU01	
Priority:	High	
Source:	Requirement	
Primary Business Actor:	Potential Member / Customer	
Other Participating Actors:	Database (Receives Member's Log In data)	
Other Interested Stakeholders:	System Owners	
Description:	This use case describes the event of a person signing up to become a new Member. The Member will input a username and a password, which will be verified to not already be in use, and once verified, stored in a database, allowing the Member to sign up and be able to login to the system.	
Precondition:	None, this is the first step needed to be taken by anyone wishing to use the system.	
Trigger:	This use case is initiated when a user wishes to sign up, and then clicks "Sign up"	
Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The user clicks the "sign up" button. **Step 3:** The user inputs their data into the required fields, and initiates the sign up.	**Step 2:** The system responds by displaying the "Sign up" page, where users can input data. **Step 4:** The system verifies that this data isn't already input (There aren't any duplicate usernames, etc…) before it stores this data in the database, and displaying the original "Log In or Sign up" page.
Alternate Course:	**Alt-step 2:** The user inputs invalid data, or leaves one or more input fields blank. **Alt-step 3:** The System responds by displaying an error or warning, and requests that the user re-enter the data properly.	
Conclusion:	This use case concludes when the user either: 1) Completes the sign up process	

	Or: 2) Exits the System.
Post-condition:	Assuming the user completed the sign up process successfully; the data they input will be stored in the database.
Business Rules:	➢ The username that the user enters must be unique, and will be compared to other usernames. ➢ The password may not be a blank input field.
Implementation Constraints and Specifications:	• GUI to be provided for users must be easy to navigate
Assumptions:	Database will be easily accessible, and functional.
Open Issues:	None.

By: TR Villet

Create Event Use Case Narrative

Use-Case Name:	Create Event	
Use-Case ID:	CE1	
Priority:	High	
Soure:	Create Event Requirement	
Primary Business Actor:	Registered User	
Description:	This use-case deals with the creation/scheduling of an event in OmniCal to be updated to a user's planned timetables and events.	
Precondition:	The user must be registered and logged in.	
Trigger:	This use-case is initiated when the user views a day in the planner and selects the option to create an event.	
Typical Course of Events:	**Actor Action** **Step 1:** The registered and logged-in user selects a day to view. **Step 3:** The user types out the information to be stored in the schedule and saves/creates it.	**System Response** **Step 2:** The selected day schedule is displayed. **Step 4:** The information is saved to the database (see "Send Infor to Database" use-case).
Alternate Courses:	n/a	
Conclusion:	An event has been stored in the specific day's schedule for the user.	
Post-condition:	The event must be sent to the database.	
Business Rules:	n/a	
Implementation Constraints and Specifications:	GUI to be provided for the user and a database for storing all necessary information.	
Assumptions:	The event will be stored appropriately in the database (separate use-case).	
Open Issues:	1. An event must be stored in a manner that includes a time of day. 2. It is undecided yet whether or not more than one event can be created for the same day and time.	

By: MC Erasmus

View Timetable Use Case Narrative

Use-Case Name:	View Timetable	
Use-Case ID:	VT1	
Source:	Database	
Primary Business Actor	Registered User	
Description:	The use-case send a specific request of a registered client to the database of OmniCal that then will send the clients current days activities for display.	
Precondition:	Only one week will be displayed in the view timetable option.	
Trigger:	The use-case will initiate when the log in use-case is complete and the view timetable option is selected.	
Typical Courses of Events:	**Actor Action** **Step 1:** The logged in user selects view day option.	**System Response** **Step 2:** Database loads the logged in users details sends the info to be displayed by the GUI.
Alternate Courses:	n/a	
Conclusion:	The view timetable is viewed or edited by the user.	
Post condition:	The View timetable use-case receives data from the database and the user.	
Business rules:	n/a	
Implementations Constraints and Specifications:	GUI to display the users schedule and database to store the users data.	
Assumptions:	The users schedule will be customised by the view timetable use-case and stored in the database in the appropriate users data.	
Open issues:	If the view timetable only displays one day at a time the users is going to take a long time to edit the schedule to there needs.	

By: R Du Plessis

If Admin Use Case Narrative

Use-Case Name:	If Admin	
Use-Case ID:	IU3	
Priority:	High	
Soure:	Admin Selection Requirement	
Primary Business Actor:	Registered User	
Description:	This use-case describes the system response to a user signing in as admin.	
Precondition:	The user must be registered.	
Trigger:	This use-case is initiated when the user logs in as admin .	
Typical Course of Events:	**Actor Action** The user chooses to log in as admin and enters the correct password.	**System Response** If entered password is valid, the user is logged into the system with admin rights.
Alternate Courses:	User can log in as an employee or student.	
Conclusion:	Proceeds to / enables the admin-related use-cases.	
Post-condition:	Only admin-related functions are enabled.	
Business Rules:	n/a	
Implementation Constraints and Specifications:	Interface and functions must be made appropriate to the user type.	
Assumptions:	The registered user type and details are stored correctly in the database.	

By: MC Erasmus

if Student Use Case Narrative

Use-Case Name:	If Student	
Use-Case ID:	IU1	
Priority:	High	
Soure:	Student Selection Requirement	
Primary Business Actor:	Registered User	
Description:	This use-case describes the system response to a user signing in as a student.	
Precondition:	The user must be registered.	
Trigger:	This use-case is initiated when the user logs in as student.	
Typical Course of Events:	**Actor Action** The user enters details and chooses to log in as a student.	**System Response** If entered details are valid, the user is logged into the system as a student and student-appropriate functions are enabled.
Alternate Courses:	User can log in as an employee or admin.	
Conclusion:	Proceeds to / enables the student-related use-cases.	
Post-condition:	Only student-related functions are enabled.	
Business Rules:	n/a	
Implementation Constraints and Specifications:	Interface and functions must be made appropriate to the user type.	
Assumptions:	The registered user type and details are stored correctly in the database.	

By: MC Erasmus

if worker (General user) Use Case Narrative

Use-Case Name:	If Worker (General User)	
Use-Case ID:	IU2	
Priority:	High	
Source:	User Selection Requirement	
Primary Business Actor:	Registered User	
Description:	This use-case describes the system response to a user signing in as an employee.	
Precondition:	The user must be registered.	
Trigger:	This use-case is initiated when the user logs in as employee .	
Typical Course of Events:	**Actor Action** The user enters details and chooses to log in as an employee .	**System Response** If entered details are valid, the user is logged into the system as an employee and employee-appropriate functions are enabled.
Alternate Courses:	User can log in as a student or admin.	
Conclusion:	Proceeds to / enables the employee-related use-cases.	
Post-condition:	Only employee-related functions are enabled.	
Business Rules:	n/a	
Implementation Constraints and Specifications:	Interface and functions must be made appropriate to the user type.	
Assumptions:	The registered user type and details are stored correctly in the database.	

By: MC Erasmus

if English Use Case Narrative

Use-Case Name:	If English	
Use-Case ID:	IEL01	
Priority:	Low	
Source:	Choose Language	
Primary Business Actor:	User	
Description:	This use-case describes the change in language if English is selected by the user	
Precondition:	n/a	
Trigger:	This use-case is initiated when the user chooses English as their preferred language .	
Typical Course of Events:	**Actor Action** The user chooses English as a language.	**System Response** The system translates the information to English
Alternate Courses:	If Afrikaans If Zulu If Sesotho	
Conclusion:	Translates all data to English	
Post-condition:	n/a	
Business Rules:	n/a	
Implementation Constraints and Specifications:	n/a	
Assumptions:	The program is running in a language which is not English	

By: J Muller

if Afrikaans Use Case Narrative

Use-Case Name:	If Afrikaans	
Use-Case ID:	IAL01	
Priority:	Low	
Source:	Choose Language	
Primary Business Actor:	User	
Description:	This use-case describes the change in language if Afrikaans is selected by the user	
Precondition:	n/a	
Trigger:	This use-case is initiated when the user chooses Afrikaans as their preferred language .	
Typical Course of Events:	**Actor Action** The user chooses Afrikaans as a language.	**System Response** The system translates the information to Afrikaans
Alternate Courses:	If English If Zulu If Sesotho	
Conclusion:	Translates all data to Afrikaans	
Post-condition:	n/a	
Business Rules:	n/a	
Implementation Constraints and Specifications:	n/a	
Assumptions:	The program is running in a language which is not Afrikaans	

By: J Muller

if Zulu Use Case Narrative

Use-Case Name:	If Zulu	
Use-Case ID:	IZL01	
Priority:	Low	
Source:	Choose Language	
Primary Business Actor:	User	
Description:	This use-case describes the change in language if Zulu is selected by the user	
Precondition:	n/a	
Trigger:	This use-case is initiated when the user chooses Zulu as their preferred language .	
Typical Course of Events:	**Actor Action** The user chooses Zulu as a language.	**System Response** The system translates the information to Zulu
Alternate Courses:	If Afrikaans If English If Sesotho	
Conclusion:	Translates all data to Zulu	
Post-condition:	n/a	
Business Rules:	n/a	
Implementation Constraints and Specifications:	n/a	
Assumptions:	The program is running in a language which is not Zulu	

By: J Muller

if Sesotho Use Case Narrative

Use-Case Name:	If Sesotho	
Use-Case ID:	ISL01	
Priority:	Low	
Source:	Choose Language	
Primary Business Actor:	User	
Description:	This use-case describes the change in language if Sesotho is selected by the user	
Precondition:	n/a	
Trigger:	This use-case is initiated when the user chooses Sesotho as their preferred language .	
Typical Course of Events:	**Actor Action** The user chooses Sesotho as a language.	**System Response** The system translates the information to Sesotho
Alternate Courses:	If Afrikaans If Zulu If English	
Conclusion:	Translates all data to Sesotho	
Post-condition:	n/a	
Business Rules:	n/a	
Implementation Constraints and Specifications:	n/a	
Assumptions:	The program is running in a language which is not Sesotho	

By: J Muller

Help file Use Case Narrative

Use-Case Name:	Help	
Use-Case ID:	HP1	
Source:	Database	
Primary Business Actor	Registered User	
Description:	The use-case send a specific request to the database of OmniCal that then will send a list of instructions of what to do in certain scenarios for the user read and understand.	
Precondition:	The help option can be accessed from any place in the system.	
Trigger:	The use-case will initiate when the help button is selected.	
Typical Courses of Events:	**Actor Action** **Step 1:** The logged in user selects the help button.	**System Response** **Step 2:** Database receives a help request from the user. **Step 3:** The database sends the list of information and scenarios to be displayed in a message box.
Alternate Courses:	n/a	
Conclusion:	The help option is available for the user if needed on any place in the system.	
Post condition:	The Help use-case receives input from the user.	
Business rules:	n/a	
Implementations Constraints and Specifications:	Input by the user processing by the database and output is a message box.	
Assumptions:	The help option is implemented to help the user better understand the system and make the user use the system with ease.	
Open issues:	The help option is optional for the user. Some users will try to figure problems by the self without the he help option.	

By: R Du Plooy

Close Program Use Case Narrative

Use-Case Name:	Close	
Use-Case ID:	CL1	
Source:	System Function	
Primary Business Actor	Registered User	
Description:	The use-case send a specific request to the running program that's OmniCal that then will implement the save use-case followed log out use-case and then will close the program.	
Precondition:	The close option can be accessed from any place in the system.	
Trigger:	The use-case will initiate when the close button is selected.	
Typical Courses of Events:	**Actor Action** **Step 1:** The logged in user selects the close button. **Step 3:** the user selects to save the changes or not.	**System Response** **Step 2:** The system asks to save the changes. **Step 4:** The changes are send to the database or the log out use-case is implemented followed by the close use-case.
Alternate Courses:	n/a	
Conclusion:	The close option is available for the user if needed on any place in the system to close the program quick.	
Post condition:	The close use-case receives input from the user.	
Business rules:	n/a	
Implementations Constraints and Specifications:	Input by the user implements the save use-case that will then sent the changes and the implement the close use-case.	
Assumptions:	The close option is implemented to help the user close the program quick and save changes made if necessary.	
Open issues:	n/a	

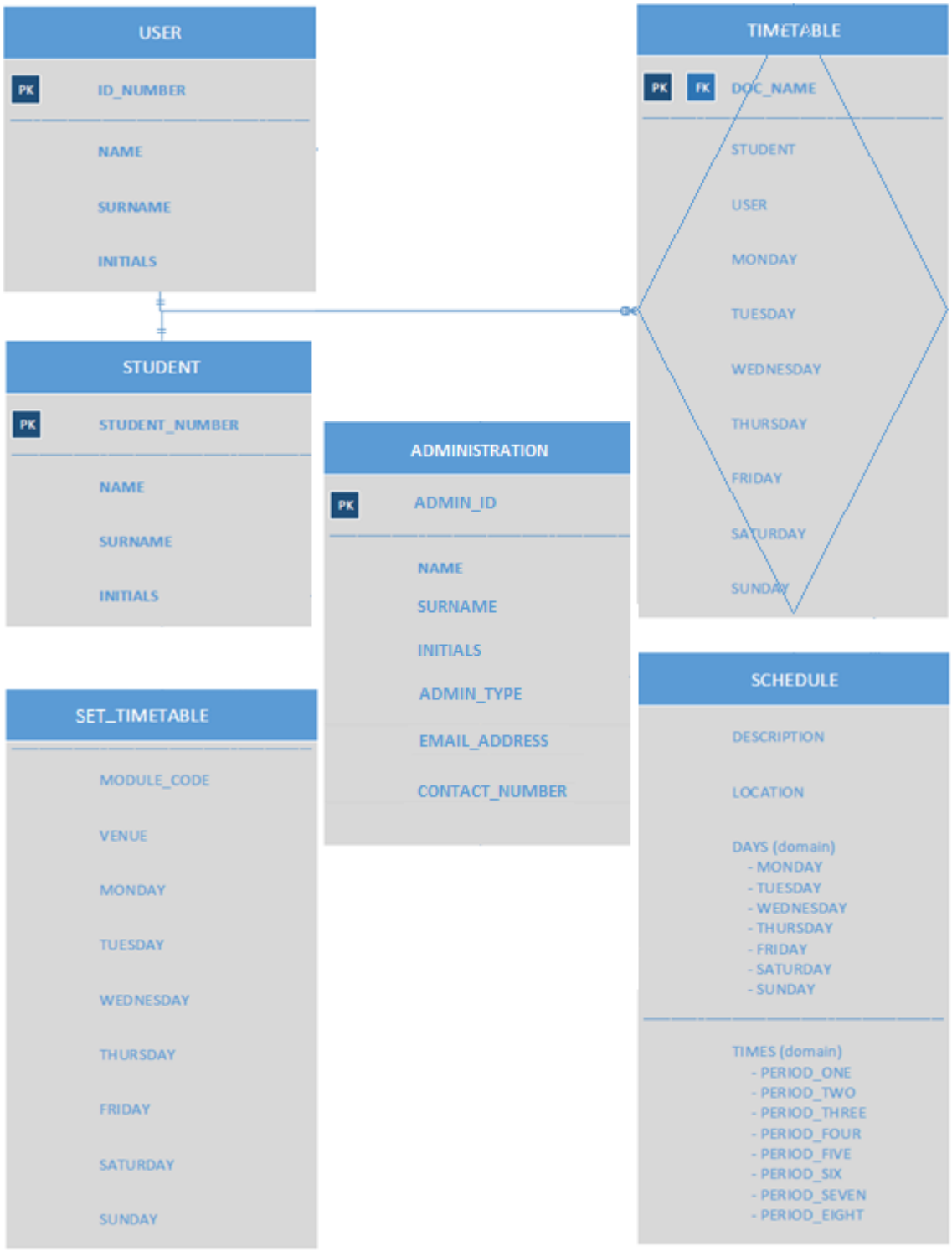
By: R Du Plooy

Save Use Case Narrative

Use-Case Name:	save	
Use-Case ID:	SV1	
Source:	System function	
Primary Business Actor	Registered User	
Description:	The use-case sends a specific request to the database of OmniCal that then will save all the changes that was made in the view day use-case	
Precondition:	The save option can be used if one or more changes has been made .	
Trigger:	The use-case will initiate when the help button is selected.	
Typical Courses of Events:	**Actor Action** **Step 1:** The logged in user selects the save button.	**System Response** **Step 2:** Database receives a save request from the user and saves the changes that was made in the view day use-case.
Alternate Courses:	The close option will implement the save use-case as well.	
Conclusion:	The save option is available in two ways for the user the save button or the close button.	
Post condition:	The save use-case receives input from the user or the close use-case.	
Business rules:	n/a	
Implementations Constraints and Specifications:	Input by the user or close use-case processing by the database and output saves the changes to the database.	
Assumptions:	The save option is implemented in two ways to make sure the user does not lose any changes made.	
Open issues:	User can save the wrong data by accident.	

By: R Du Plooy

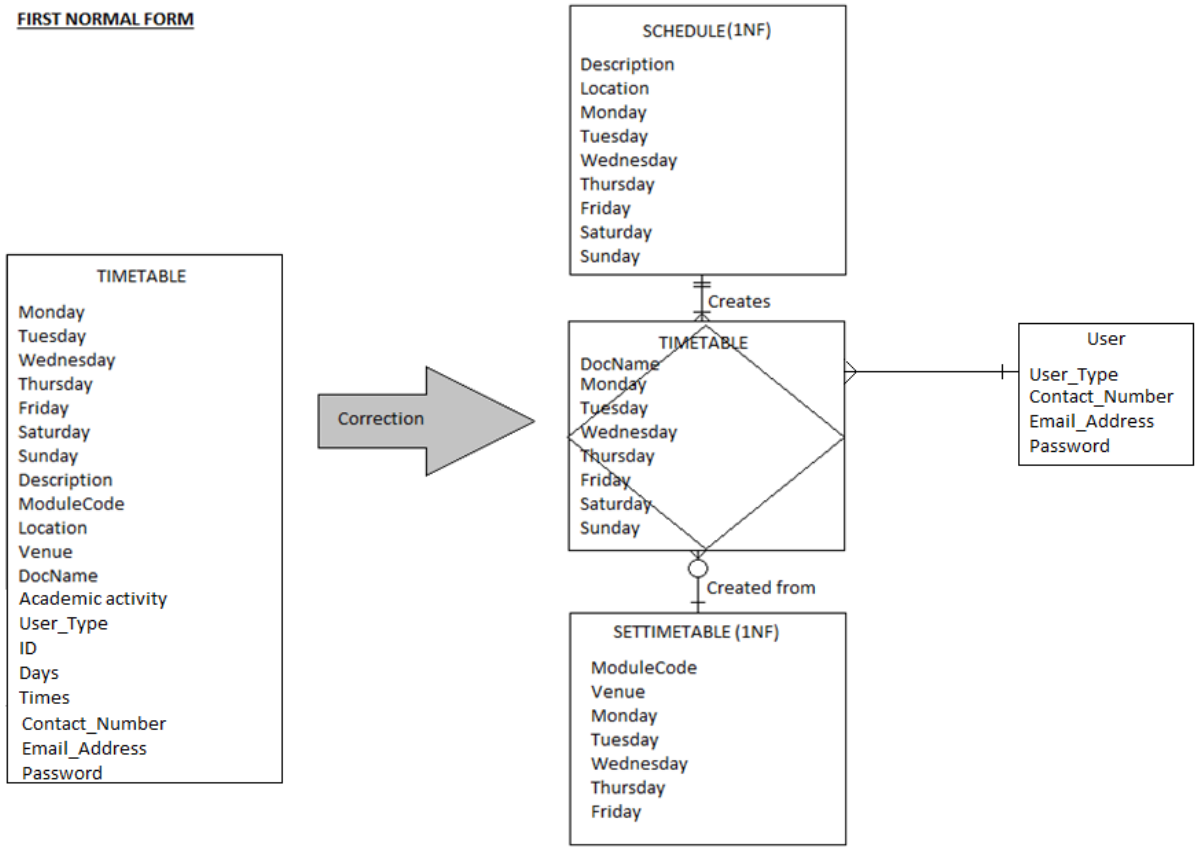
INITIAL ENTITY-RELATIONSHIP DIAGRAM



By: Everyone in the group

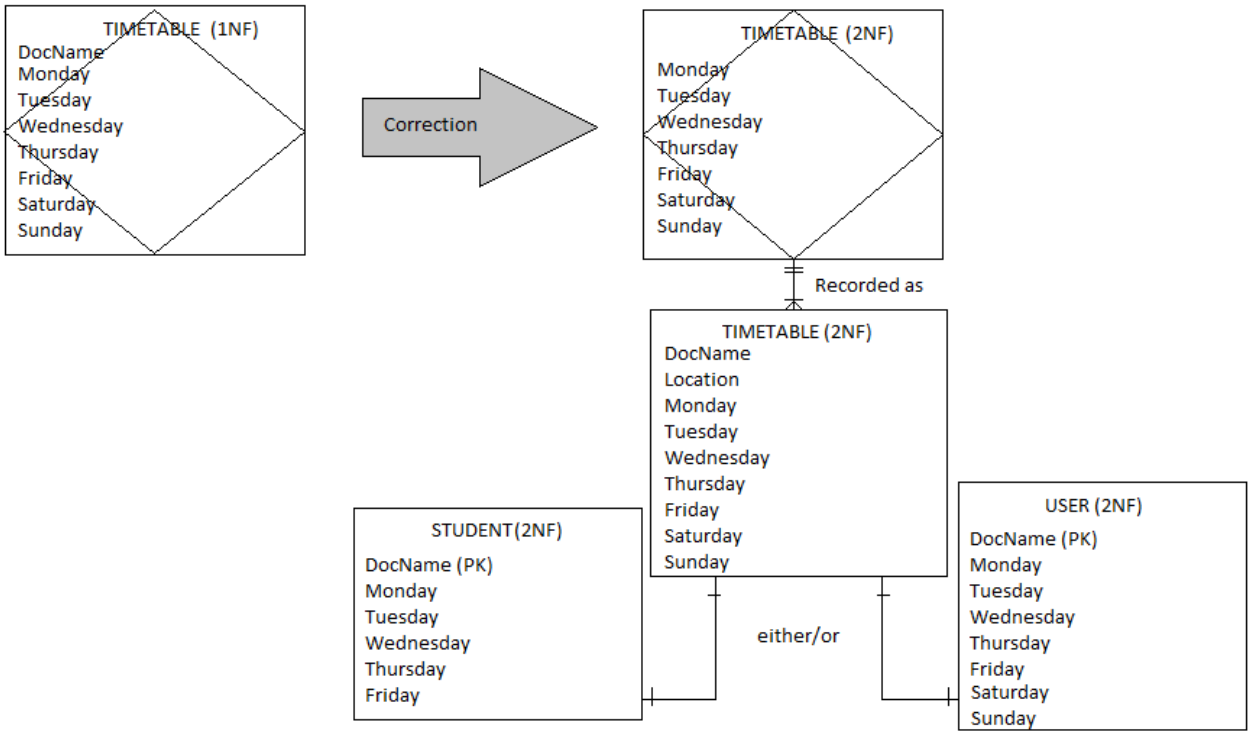
NORMALISATION

First Normal Form

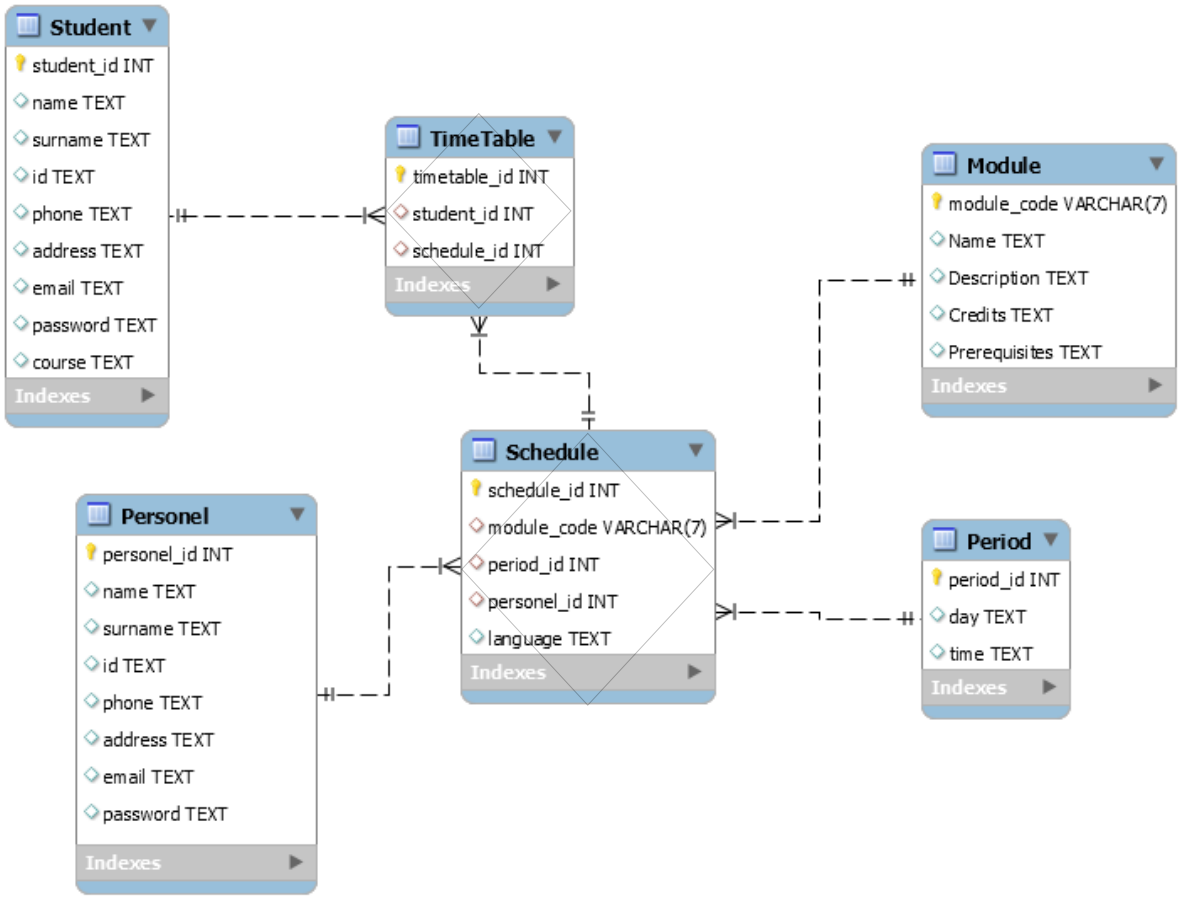


Second Normal Form

SECOND NORMAL FORM



Third Normal Form



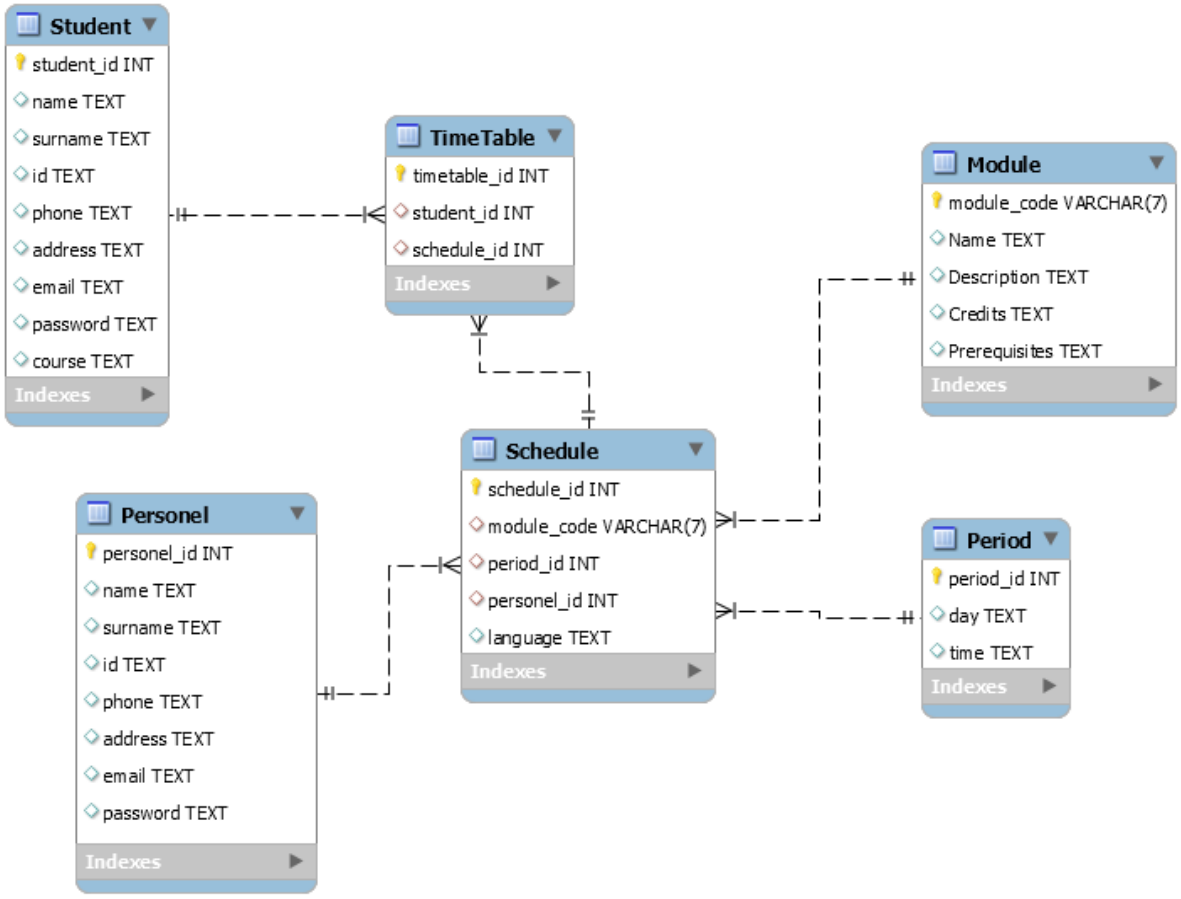
By: J Muller

DATABASE TABLE

Entity		Data Type	Required	Validation Text	Length
STUDENT					
(PK)	student_ID	INT	Yes		8
	name	TEXT	Yes		45
	surname	TEXT	Yes		45
	id	TEXT	Yes		13
	phone	TEXT	Yes		10
	address	TEXT	Yes		45
	email	TEXT	Yes		45
	course	TEXT	Yes		30
	password	TEXT	Yes		12
	year	TEXT	Yes	"First", "Second", "Third"	6
TIMETABLE					
(PK)	timetable_id	INT	Yes		10
	student _id	INT	Yes		8
	schedule_id	INT	Yes		10
MODULE					
(PK)	module_code	TEXT	Yes		7
	name	TEXT	Yes		20
	description	TEXT	Yes		45
	credits	TEXT	Yes		2
	prerequisites	TEXT	Yes		45
PERIOD					
(PK)	period_id	INT	Yes		2
	day	TEXT	Yes	"Monday", "Tuesday", "Wednesday", "Thursday", "Friday"	9
	time	TEXT	Yes	"08:00-09:30", "09:30-11","11:00-12:30", "12:30-14:00", "14:00-15:30", "15:30-17:00"	11
SCHEDULE					
(PK)	schedule_id	INT	Yes		10
	module_code	TEXT	Yes		7
	period_id	TEXT	Yes		2
	personnel_id	TEXT	Yes		8
	language_id	TEXT	Yes		10
PERSONNEL					
(PK)	personnel_id	TEXT	Yes		8
	name	TEXT	Yes		45
	surname	TEXT	Yes		45
	id	TEXT	Yes		13
	phone	TEXT	Yes		10
	address	TEXT	Yes		45
	email	TEXT	Yes		45
	password	TEXT	Yes		12

By: J Muller

FINAL ENTITY RELATIONSHIP DIAGRAM



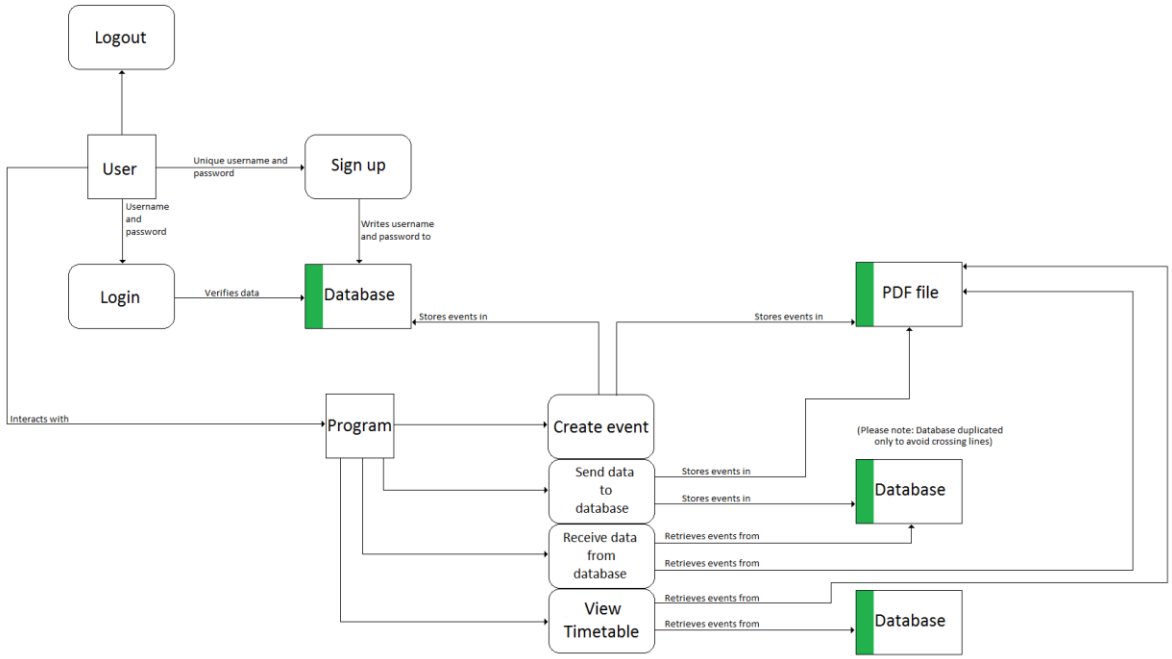
CRUD MATRIX

OMNICAL CRUD MATRIX (DATA-TO-USER)				
Entity and attributes	User	Student	Other	Lecturer	Admin
TimeTable		INDV	INDV	INDV	ALL
TimeTable_ID		X	X	X	ALL
Student ID		CS	CS	X	ALL
Schedule_ID		X	X	X	ALL
Student		INDV	INDV	X	ALL
Student_ID		CS	CS	X	ALL
Name		CS	CS	X	ALL
Surname		CS	CS	X	ALL
Phone		CSU	CSU	X	ALL
Address		CSU	CSU	X	ALL
Email		CSU	CSU	X	ALL
Password		CSU	CSU	X	ALL
ID_Number		CS	CS	X	ALL
Schedule		X	X	INDV	ALL
Schedule_ID		X	X	SRUD	ALL
Module_Code		X	X	SRUD	ALL
Period_ID		X	X	SRUD	ALL
Personal_ID		X	X	SRUD	ALL
ScheduleCol		X	X	SRUD	ALL
Personel		X	X	INDV	ALL
Personel_ID		X	X	CRUDS	ALL
Name		X	X	CRUDS	ALL
Surname		X	X	CRUDS	ALL
ID_Number		X	X	CRUDS	ALL
Phone		X	X	CRUDS	ALL
Address		X	X	CRUDS	ALL
Email		X	X	CRUDS	ALL
Password		X	X	CRUDS	ALL
Period		INDV	INDV	INDV	ALL
Period_ID		R	R	R	ALL
Days		R	R	R	ALL
Time		R	R	R	ALL
Module		X	X	INDV	ALL
Module_Code		X	X	SUD	ALL
Name		X	X	SUD	ALL
Description		X	X	SUD	ALL
Credits		X	X	SUD	ALL
Preresiquites		X	X	SUD	ALL

X = None C = Create R = Read
U = Update D = Delete S = Submit

By: TB Bensch & J Muller

DATA FLOW DIAGRAM

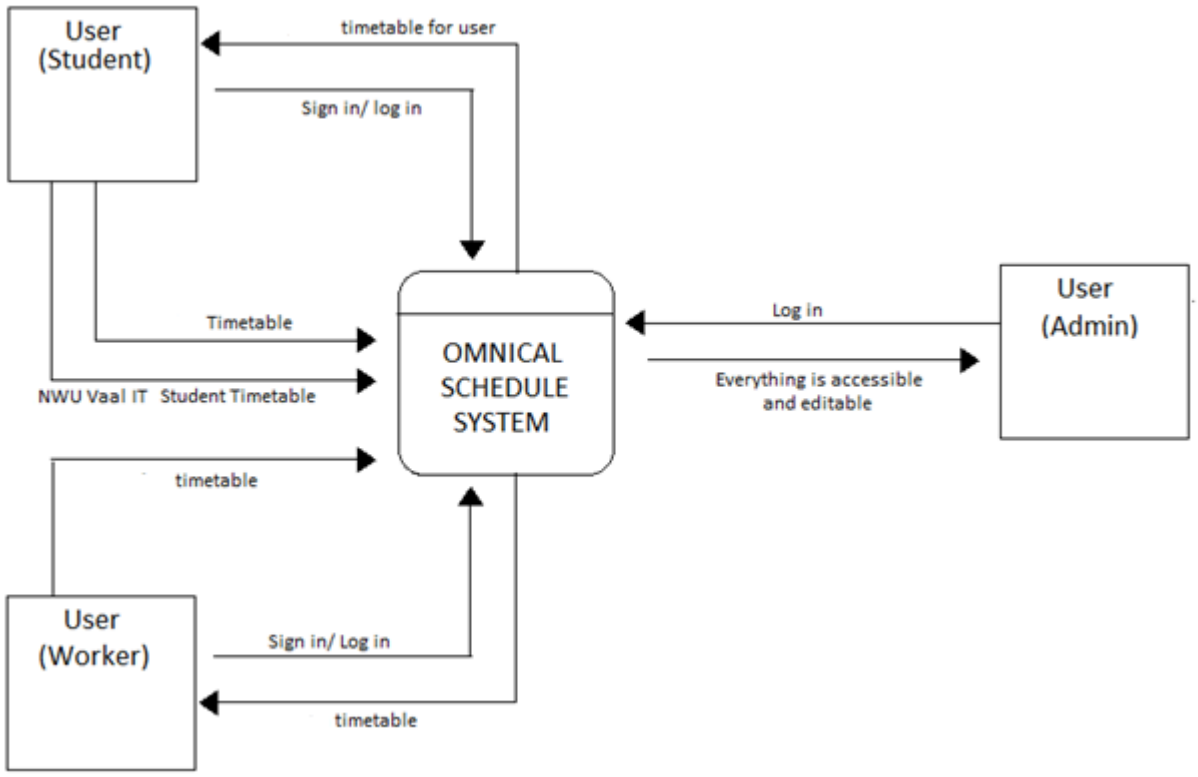


By: TR Villet

CONTEXT DATA FLOW DIAGRAM (Step 1)

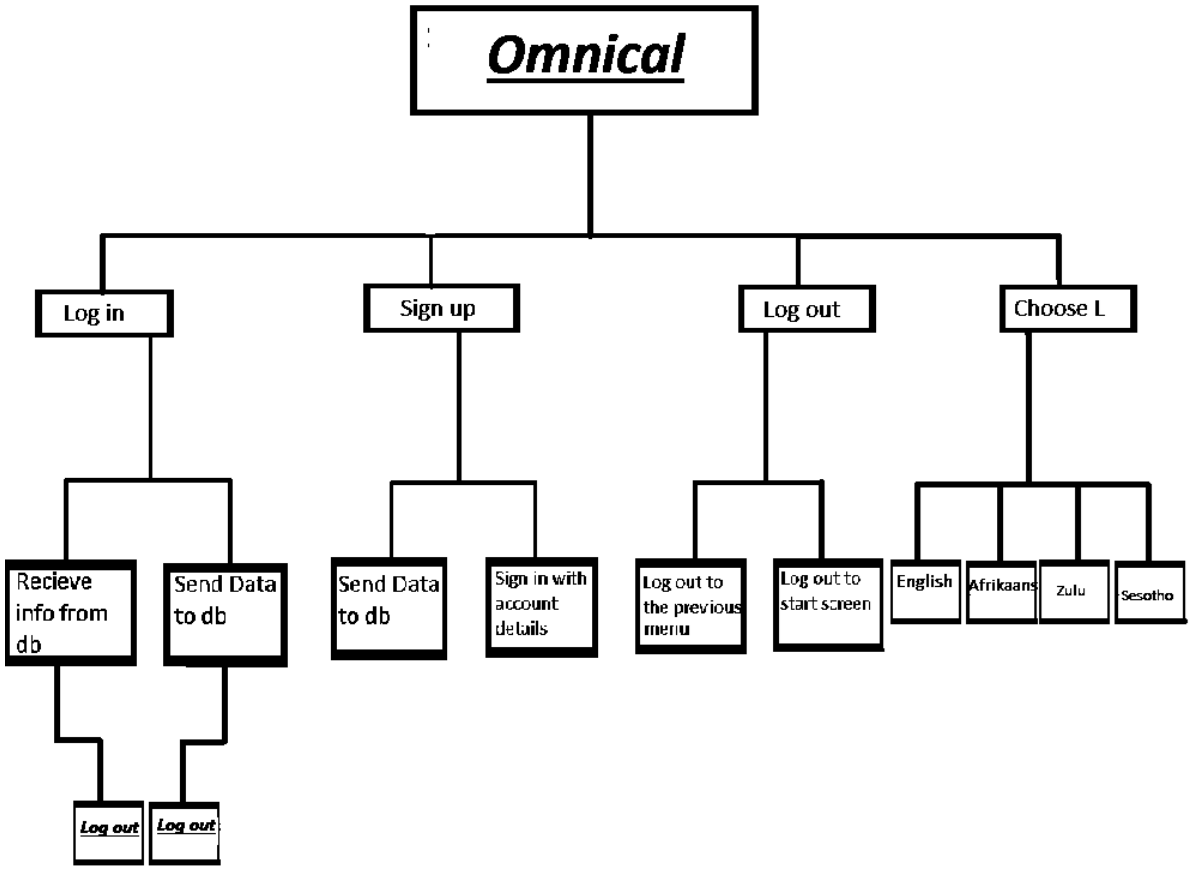
OmniCal Context Data Flow Diagram

This context Data Flow Diagram shows the basic flow of data between the system and the external users.



(Discussed by group) Compiled By: J Muller

FUNCTIONAL DECOMPOSITION DIAGRAM AND EVENT HANDLERS (Step 2 and 4)



(Discussed by group) Compiled By: R Hammond

USE CASE LIST (Step 3)

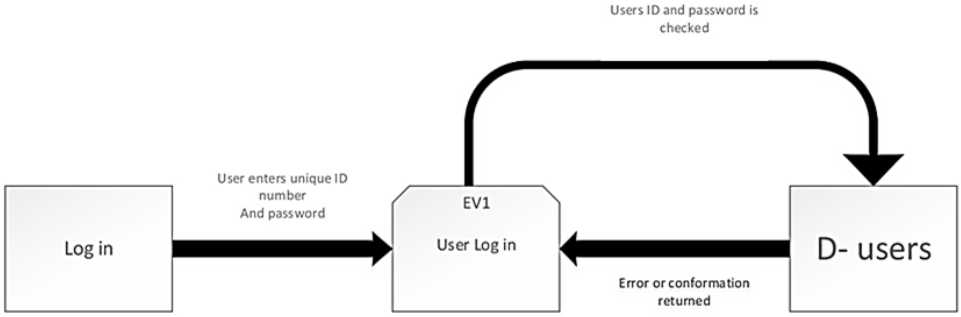
Use case list:

Event	Response
Receive information from database	Database sends information to where it is needed
Send information to database	Database receives information from where it was created, changed or used
Log in	Data is now accessible and editable
Log out	Data is no longer accessible and editable
Sign up	Information is created in order for log in to occur
Create event	Data is created to be sent to the database.
View timetable	Data is accessed and can be viewed
If admin	All data is accessible and editable
If student	Data for only that student can be accessed and edited
If worker	Data for only that worker can be accessed and edited
Help file	No data used
If English	Data is translated to English
If Afrikaans	Data is translated to Afrikaans
If Zulu	Data is translated to Zulu
If Sesotho	Data is translated to Sesotho
Close program	All data is released

(Discussed by group) Compiled By: J Muller

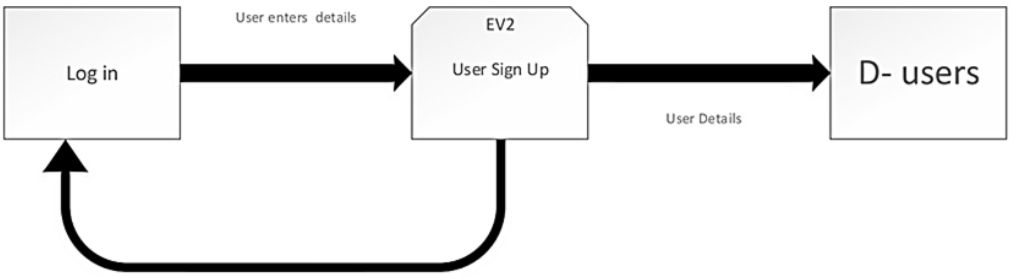
INDIVIDUAL EVENT HANDLERS (Step 5)

EVENT 1: User log in



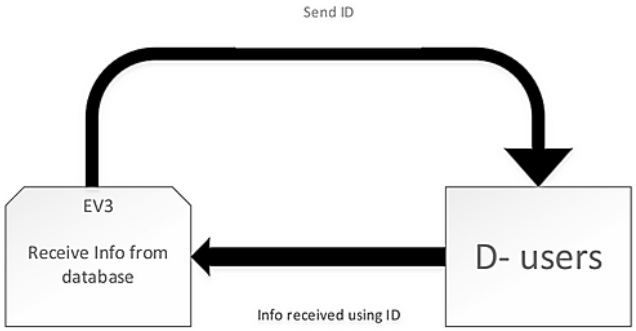
By: TB Bensch

EVENT 2: User sign up



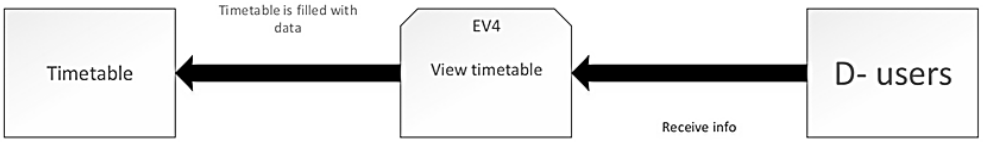
By: TB Bensch

EVENT 3: Receive information from database



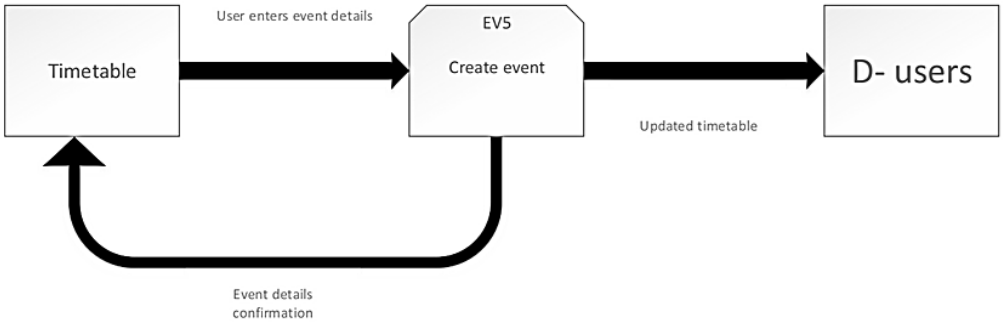
By: TB Bensch

EVENT 4: View Timetable



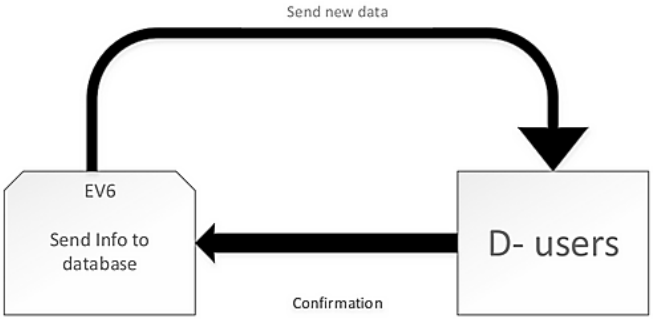
By: TB Bensch

EVENT 5: Create event



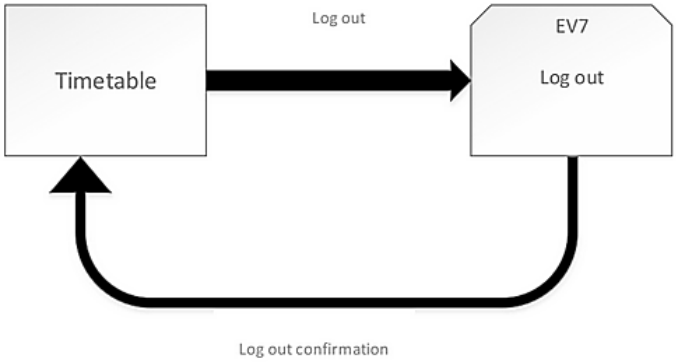
By: TB Bensch

EVENT 6: Send information to database



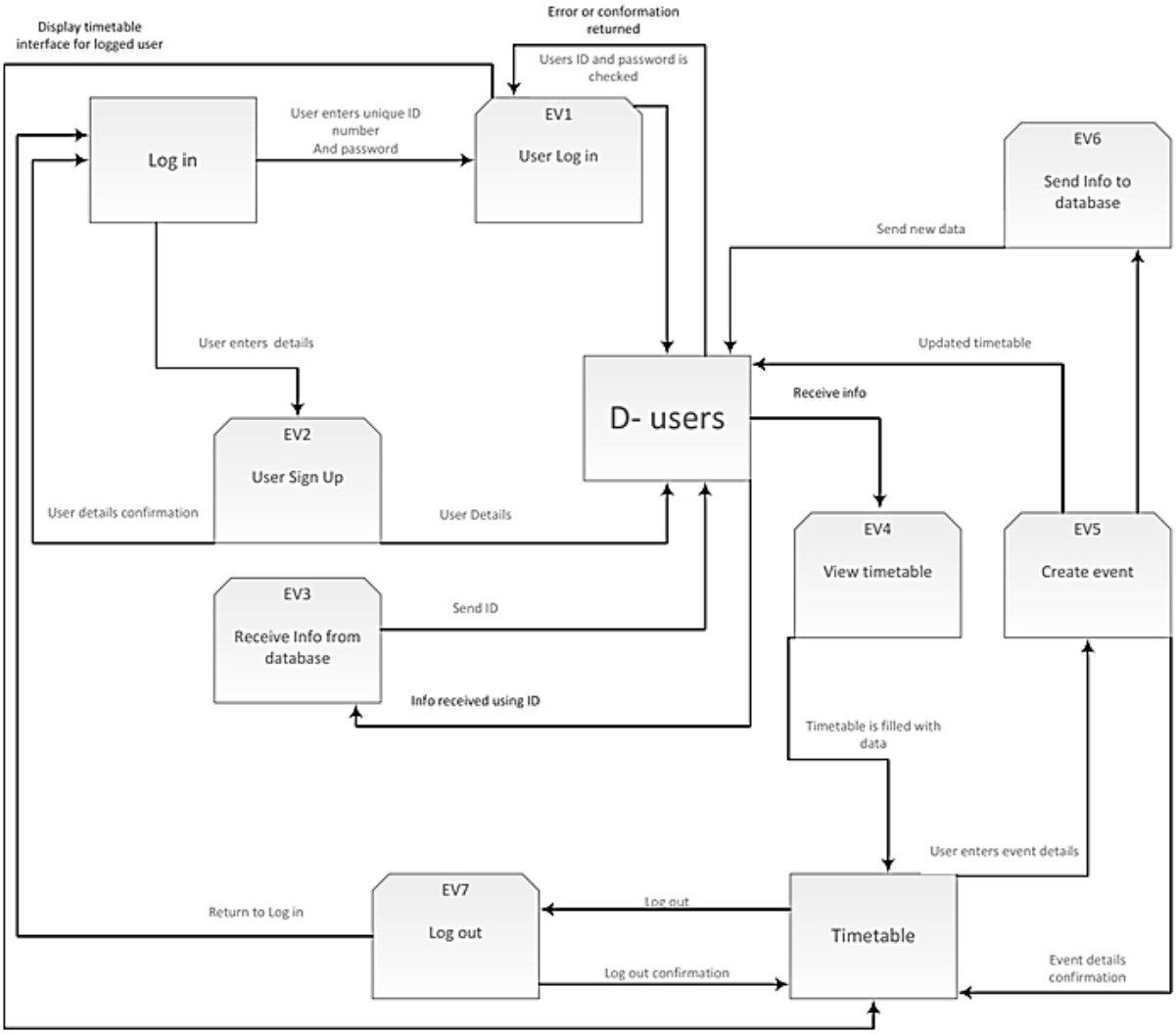
By: TB Bensch

EVENT 7: Log out



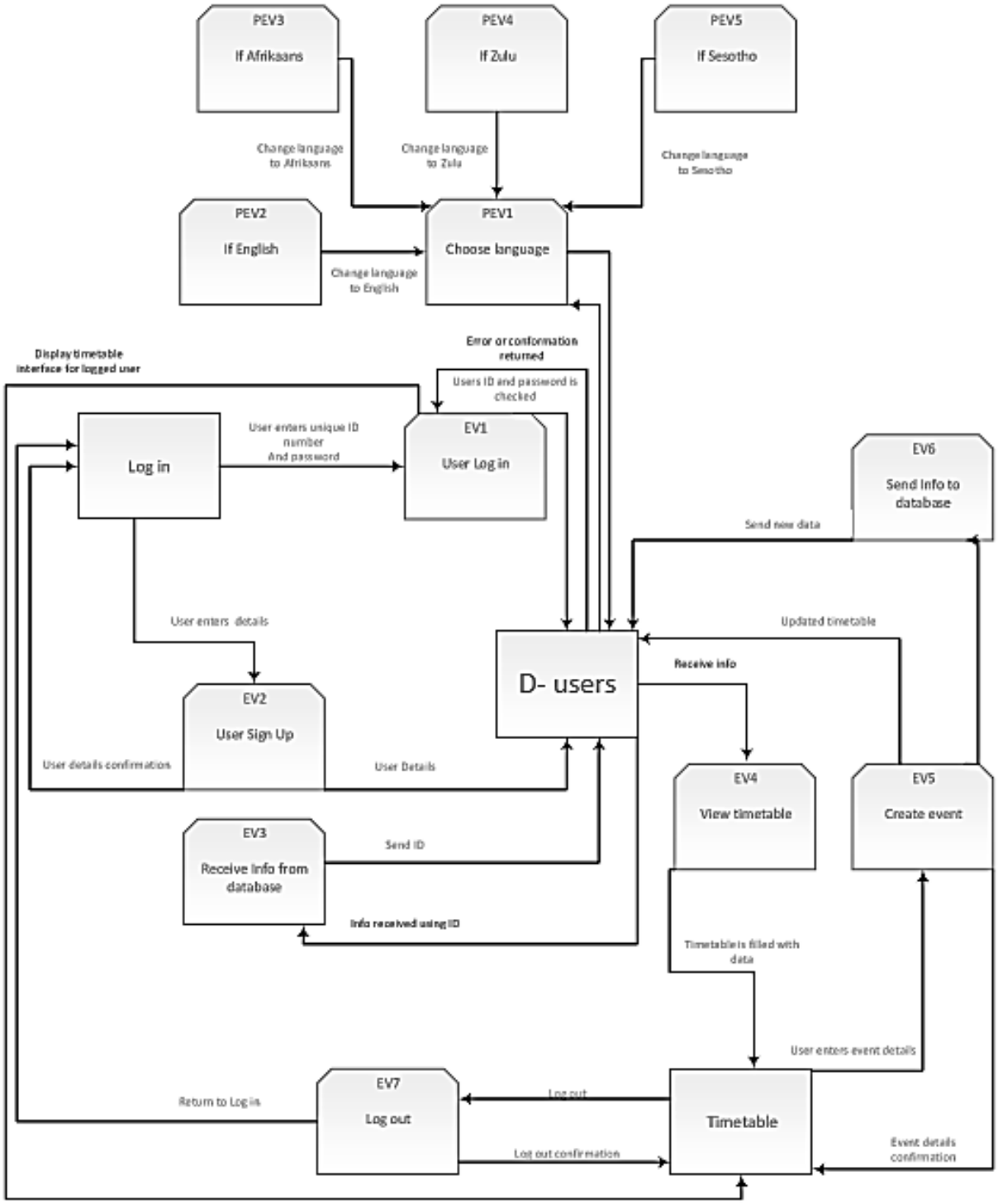
By: TB Bensch

EVENT DATA FLOW DIAGRAM (Step 6)



(Discussed by the group) Compiled By: MC Erasmus

PRIMITIVE DATA FLOW DIAGRAM (Step 7)



Compiled By: MC Erasmus and J Muller

LOGICAL PROCESSES (Step 8)

1. Admin of the system can only log in with an admin password.
2. A client must sign up before using the system/log in to the system.
3. Can't submit an empty schedule.
4. A user can't view an empty schedule.
5. Every NWU IT student have to log in before any changes can be made.
6. Every NWU IT student has a pre scheduled schedule of the IT course.
7. Workers that sign up will have an empty schedule that they can edit.
8. Every client must select a preferable language to use the system.
9. A user agreement must be read and accepted if they want to use the system.

By: R Du Plooy

DATA STRUCTURES (Step 9)

- Arrays: 2D arrays will be used to store the records from the database in order to use them in the schedule grid, as well as to extract information from the schedule grid.
- Records will be used in the database to store information across various fields.
- Fields will be used to store data in groups of specific data types and specifications across records.
- OmniCal will make use of 6 tables to store these fields and records.
    - 3 tables will be dedicated to user log in information
    - The other 3 will be used for the purpose of constructing and/or viewing the schedule.
- These tables will be stored in a database.
- 4 textfiles will be used to store the translated text of English, Afrikaans, Zulu and Sesotho.
- A pdf per user will be used to save the final schedule which can then be downloaded and used by the user.

By: J Muller

PROCESS MODELLING (Steps 1-9)

Compiled By: J Muller

FEASIBILITY ANALYSIS

OPERATIONAL FEASIBILITY

The system help the user better plan their day to day activities. The scope of the system is reached , because the system does everything that was described in the written report.

PIECES Framework

- **Performance** - Does the system provide an easy and effective result for the users?
  The system provides an improved day to day planning without struggle.

- **Information** - Does the system private useful and accurate information?
  The system provide a better validation of the user's data and is presented in a well formatted display.

- **Economics** – Is the system cost-effective?
  We made sure that the time and money is spend as effectively as possible to ensure that the system is cost effective.

- **Control –** Does the system provide the user with privacy and security?
  The system uses a Staff or Student number as identification and a password to gain access to the systems functions.

- **Efficiency –** Is the Flow of events and data storage implemented so that it is fully optimized?
  The flow of event id designed so that the user can access and uses the system in a fast and effected manner. The database is designed to reduce redundancy for safe data storage and transfer.

- **Service –** Does the system provide a useful service?
  The system provides the user with a quick and easy way to plan out his or her day.
  The system can also be used in a option of multiple languages if the user wants the system in their preferred language.

  With the PIECE Framework it is show that the system is operational and the problem is worth solving to complete the system.

By: R Du Plooy

CULTURAL (OR POLITICAL) FEASIBILITY

OmniCal is a simple system for timetable planning. Thus, there are very little, if any at all, cultural or political consequences that may cause opposition to the system. It is intended for the academic environment and should be well received as students and lecturers need some way to keep track of times, and OmniCal provides just that.

Questions that address this concern:

- *Does management support this system?*
  Yes.
- *How do end users feel about their role in the new system?*
  The end users' role is simply to use the system. It as an available tool which many may want to use, but don't have to use. As such, there are no hostilities from users (if someone does not want to use the system, they simply don't; it will not affect them in any way).
- *Who may resist or not use the system? Can this be a problem to overcome? If so, how?*
  As stated previously, there should be little or no resistance. However, many may not use it as they have other means of keeping record of schedules (such as physical books). Unfortunately, there is no way of overcoming this problem. We could market and persuade the importance for using the system, but ultimately it relies on the user's chosen method of scheduling.
- *How will the working environment of the end users change?*
  If used correctly, the system can enhance the users' ability to be on time and not forget important events.

After analyzing OmniCal through the scope of Cultural feasibility, OmniCal is shown to be feasible.

By: MC Erasmus

TECHNICAL FEASIBILITY

Questions posed:

1. Is the proposed technology or solution practical?
2. Do we currently process the necessary technology?
3. Do we process the necessary technical expertise?

Answers:

1. Yes this proposed solution is practical
2. Yes we do process the necessary technology
3. Yes we do process the necessary technical expertise.

Therefore, OmiCal can be seen as feasible.

By: J Muller

SCHEDULE FEASIBILITTY

Schedule Feasibility is a measure of how reasonable the project timetable is.

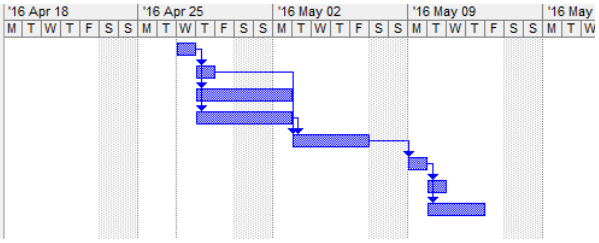
How do we gauge the feasibility of the schedule?

We gauge the feasibility of the schedule by analysing the deadlines and the project schedule, and then asking some critical questions:

Are the project deadlines reasonable?
Are the project deadlines 'mandatory'?
Are the project deadlines 'desirable'?
Do the deadlines allow some room for error?
Can the project deadlines be amended?
Should the project deadlines be amended?

The deadlines of the OmniCal project are as follows:

	Task Name	Duration	Start	Finish	Predecessors
1	Planning	1 day	Wed 16-04-27	Wed 16-04-27	
4	Design Database	1 day	Thu 16-04-28	Thu 16-04-28	1
2	Documentation	3 days	Thu 16-04-28	Mon 16-05-02	1
3	Design GUI	3 days	Thu 16-04-28	Mon 16-05-02	1
5	Coding	4 days	Tue 16-05-03	Fri 16-05-06	3;4
6	Testing	1 day	Mon 16-05-09	Mon 16-05-09	5
7	Feedback	1 day	Tue 16-05-10	Tue 16-05-10	6
8	Improvements	3 days?	Tue 16-05-10	Thu 16-05-12	6



The analysis of the project schedule and the deadlines implemented in the project schedule, shows that the deadlines are quite reasonable, leaving enough time for each member to complete their assigned work, ensuring both quality and timeliness.

The deadlines are of the "desirable" type, meaning that they may be changed. These deadlines also leave enough room for error while still attaining the desired completion time, and can be amended if the need to do so arises.

ECONOMIC FEASIBILITY

It can be seen in the cost benefit analysis that the project is cost effective.

LEGAL FEASIBILITY

There are no foreseeable problems regarding the legal field.

COST-BENEFIT ANALYSIS

Development costs:	
Personnel	
- 1 Systems Analyst (24 hours at R100.00/hr)	R2400.00
- 1 Project Manager (36 hours at R100.00/hr)	R3600.00
- 1 GUI Designer (6 hours at R100.00/hr)	R600.00
- 1 Database Designer/Expert (6 hours at R100/hr)	R600.00
- 2 Programmers (24 hours at R100.00/hr each)	R4800.00
Expenses	
Internet usage during development	R59.00
Software	
- Microsoft Visual Studios 2015 (Free)	R0.00
- Visio Pro for Office 365 (1 month at R266,60/m)	R266.60
TOTAL DEVELOPMENT COSTS:	*R12325.60*
Projected Annual Operating Costs:	
Personnel	
- Programmer/analyst(20 hours at R100.00/hr)	R20000.00
TOTAL PROJECTED ANNUAL COSTS:	R20000.00
TOTAL COSTS	**R32325.60**
Personnel Benefits from OmniCal:	
- 1 Project manager	R7950.00
- 1 System analyst	R7950.00
- 2 System designers (R7950.00 each)	R15900.00
- 2 System builders (R7950.00 each)	R15900.00
TOTAL FORESEEABLE PERSONNEL BENEFITS	*R47700.00*
Other comprehensive income:	
- Use of full version (R10 each)	R1000.00
- Donations	R5000.00
TOTAL INCOME	*R6000.00*
TOTAL BENEFITS	**R53700.00**

Cost Benefit Ratio

R 32 325.60 : R 53 700.00
R1 : R 1.66

Benefits exceed costs by R 21 375.40

By: J Muller & MC Erasmus

CANDIDATE ANALYSIS MATRIX

Characteristics	**Candidate 1** (Develop with Java)	**Candidate 2** (Develop with C#)	**Candidate 3** (Develop with Delphi)
Portion of System Computerized	Entirety of OmniCal will be self-developed and computerized.	Entirety of OmniCal will be self-developed and computerized.	Entirety of OmniCal will be self-developed and computerized.
Benefits	Wide range of methods and classes allows for greater diversity and freedom in programming.	Easy to design GUI. Easy to link to database.	Easy drag-and-drop GUI designer. Simple coding.
Disadvantages	Time-consuming to design GUI; linking to database is more complex.	Limited look-and-feel customization.	Limited look-and-feel customization. Few experienced coders
Software Tools Needed	BlueJ and JDK	Microsoft Visual Studios C#	Delphi programming package
Application Software	Custom solution	Custom solution	Custom solution
Method of Data Processing	Client	Client	Client
Output Devices and Implications	Computer with adequate hardware/software. Monitor.	Computer with adequate hardware/software. Monitor.	Computer with adequate hardware/software. Monitor.
Input Devices and Implications	Keyboard and mouse (unless touch-screen monitor).	Keyboard and mouse (unless touch-screen monitor).	Keyboard and mouse (unless touch-screen monitor).
Storage Devices and Implications	Hard-drive with sufficient storage space on client computer.	Hard-drive with sufficient storage space on client computer.	Hard-drive with sufficient storage space on client computer.

By: MC Erasmus

FEASIBILITY ANALYSIS MATRIX

Characteristics	Weight	Candidate 1	Candidate 2	Candidate 3
Description		Write application using BlueJ (Java)	Write application using Microsoft Visual Studios C#	Write application using Delphi 7
Operational feasibility	15%	Supports full functionality. *Score: 100*	Supports full functionality. *Score: 100*	Supports full functionality. *Score: 100*
Cultural feasibility	15%	No foreseeable problems. *Score: 100*	No foreseeable problems. *Score: 100*	No foreseeable problems. *Score: 100*
Technical feasibility	20%	Entirety of GUI must be coded. Slightly more flexible (allows wider variety of coding). *Score: 80*	Easy to design GUI. Programmers are more knowledgeable with how to link to database. *Score: 95*	Easy to design GUI. Programmers are more knowledgeable with how to link to database. *Score: 95*
Economic feasibility	30%	Will be longer to develop and thus more costly. *Score: 60*	Less time to develop mean reduced cost. *Score: 75*	Less time to develop mean reduced cost. *Score: 75*
Schedule feasibility	10%	Approx. 14 days *Score: 95*	Approx. 12 days *Score: 100*	Approx. 12 days *Score: 100*
Legal feasibility	10%	No foreseeable problems. *Score: 100*	No foreseeable problems. *Score: 100*	No foreseeable problems. *Score: 100*
Weighted Score	100%	**83.5**	**91.5**	**91.5**

By: J Muller & MC Erasmus

WRITTEN REPORT

Executive Level Report

OmniCal is a powerful piece of day-planning, calendar-style academic organizer software, designed with students and lecturers in mind.

Stakeholders working towards optimum user experience

OmniCal stakeholders are a dedicated team made up of six individuals striving to see OmniCal succeed. They are hardworking and have put maximum effort into this scheduling system. The team is made up of a project manager, a system analyst, two system designers and two system builders:

- Project manager : Jacqueline Muller
- System analyst : Tristin Bensch
- System designer : Tyden Villet
- System designer : Ricco Hammond
- System builder : Michael Erasmus
- System builder : Renaldo du Plooy

Throughout documentation these six individuals worked in pairs. The pairings differed from week to week depending on the processes and work that needed to be completed but it followed the principle below:

- Pair 1 : One system designer and one system builder
- Pair 2 : One system designer and one system builder
- Pair 3 : Project manager and system analyst

This concept contributed greatly to the development documentation through means of compromising perceptions of what the work should look like to make the building of the system easier for the system builders. This also enhanced communication between stakeholders and each individual learned how the other thinks, reasons and works.

Purpose of OmniCal:

OmniCal Allows the user to organize and put their time to better use, through arranging events and even organizing their timetables, with the help of this easy-to-use and easy-to-understand software. OmniCal can automatically search and develop a timetable for a NWU Vaal BSclnormationTechhnology student, any other student or lecturer through means of academic records, extra modules as well as manual entries (for example, SI or assistant classes):

- A login system will be used. The user will be able to log on and off when desired.
- As a NWU Vaal BSclNformationTechnology student, the automatic timetable option will be available for first second or third years. Students studying anything else on campus will not have that option, they will have to set up their timetable manually.
- A user-friendly, calendar-styled day planner will display a daily schedule.

- It will allow quick and easy access to all your planned events and flexible editing of all events.
- It will have an intuitive, easy-to-understand user interface.
- It will allow mainly for academic use, but is not limiting users to only university academic activities.
- It is simple and light on hardware, not requiring much processing power.
- It will allow students and lecturers to be up to date with all their planned activities, classes, tests, due-dates, appointments and other events.
- A timetable will be created on the timetable screen once the user has logged in for the first time. The program will load the desired timetable from a database and the user can add any extra classes to their timetable.
- The user will be able to edit their timetable whenever they desire so.
- OmniCal will also keep track of extra-curricular activities as well as provide suitable diets.

Benefits of Omnical for the user

- Allows for ease of access to daily schedules.
- Makes it easier for people to see what they have planned for the week so that they can spread out their work load.
- Editing will allow for users to personalize and create a timetable that feels like it's their own.
- Allows for users to plan ahead and not miss anything important in their daily lives, from classes to personal activities.
- Allowing users to edit freely and have peace of mind when it comes to planning ahead.
- Reduces the risk of incorrect input / output (GIGO principle).
- OmniCal saves tie. Due to the fact that timetables are preloaded into a database, it would require less time to set up a standard timetable.
- No difficulty setting up your timetable for classes as Omnical loads the prewritten timetable from a database and puts it into your schedule for you.

User manual

The use of OmniCal can become confusing with regards to which users have access to the automate button and which don't.

Sign up:
A sign up option is available on the home screen for a user who is not yet registered on the system after the user has selected user type. Once the user has signed up, the user has to log in.

Log In:
A home screen will appear which allows the user to pick their type of user, either for student or lecturer. A student is divided into two categories by course, a BScInformationTechnology student and a student studying anything else. The user can now create a timetable.

Create/ Edit a timetable:
If the user is a BScInformationTechnology student, the automatic timetable as well as the manual options are available else the user can only create a timetable manually. Once the timetable is created, the information is saved and the full timetable can now be viewed.

View timetable:
Upon selection of the view timetable option, the user can now view their timetable thereafter, they can save it as a PDF or edit it.

Save as PDF:
The user can make use of the "Save as PDF" option provided which will save a copy of the timetable onto the user's computer in PDF format.

User feedback on system

Strategically, the system analyst and project manager chose two users to evaluate OmniCal at two stages of development.

The first interviewee, Kholofelo Poppy Mochele, is a second year BScIT student who is familiar with the development environment. She was chosen because her experience with the development process could give an accurate insight from both a user's and a system's analyst's perspective.

The second interviewee, Lizelle De Bruin, who is not very familiar with IT perspectives. She was chosen due to her lack of knowledge about system development which would give a more accurate indication of general user feedback.

The first interview was conducted by the project manager, J Muller, before the prototyping of the system was underway. The second interview was carried out by the system analyst, TB Bensch, after the user was presented with the prototype to test.

After the four interviews were conducted, the answers from the two different users were compared to provide the best user experience for all users, those who are familiar with the environment and those who are not. It was noted that users do not want a complicated system but rather something that does and effective job in sufficient time with little struggle involved. It was also suggested that having various language options would enhance a suitable user experience. Another suggestion was that the user's schedule be saved in a document, preferably in a PDF format. A useful suggestion was that OmniCal refer to "Workers" as a "General User" to avoid any confusion.

Both interviews turned out fruitful with a few very good suggestions made and views shared from samples of very different user experiences.

Cost-benefit analysis

The cost-benefit analysis can be summarized as follows:

TOTAL COSTS: R32 325.60	TOAL BENEFITS: R53 700.00
R 32 325.60 : R 53 700.00	Benefits exceed costs by R 21 375.40
R1 : R 1.66	For every R1 cost, OmniCal returns a R1.66 benefit

Supervisory Level Report

Planning

We planed the project well, we accounted for holidays and weekends as shown in the Gantt chart. We also took necessary precautions just in case we didn't have enough time for a specific part of the project, meaning that we allocated more time but not enough to ruin the planning of the project.

Documentation

We documented everything we could,

- Use case diagrams
- use case narratives
- questionnaires
- interviews
- ERD's,
- CRUD matrixes
- Scheduling charts
- Many more
- Normalisation
- Fishbone Diagrams
- System requirements
- System development Methodologies
- PIECES framework
- Much more

We included appropriate time for documentation in planning meaning that we didn't have a shortage of time nor documentation in the end.

Design GUI

The GUI was designed to be user friendly as well as appeal to people. Trying to find the right combination was tricky but we believe that we found the best GUI that could be used with our program, it's both eye appealing and easy to use. The GUI is one of the many strengths of Omnical.

Design database

We accounted for the different actors in the project : Students, other students, Lecturers and Admin in the database. Many have access to certain areas and many do not. Such as student not having access to the password of the Lecturer. The database holds the modules, period, timetables, student info, lecturer info and all necessary admin fields.

Coding

After the database and graphical user interface have been designed, the next step is coding.

Coding is the core of the system development project; it is the physical development of a functioning system. In our system development project, coding consists of two parts, namely linking the system to the database and coding functionality of the interface.

Four days have been allocated to coding. Of those, three are dedicated to developing a working prototype with correct functionality and database usage.

The fourth day is dedicated to finalizing the prototype by handling exceptions, identifying and solving logical errors and improving interface and functionality if appropriate.

Objectives include:
-Implement appropriate objects and methods.
-Enable login page to adept to user selection, validate input and read from and write to the users part of the database.
-Enable timetable to link to database with functionality to read from and write to according to user selection and input.
-Create interface adaptability based on language selection.

Once coding is completed, the system will be fully functional.

Testing

Testing involves getting "test" users to try out the system. This is important for identifying logical errors that could be overlooked during coding, as well as evaluating user-friendliness of the interface.

During this phase, testers are required to use every aspect of the system as if for real purposes. Testers must take note of the interface, functionality, logical flow of events and ease. This is to be followed by feedback.

It is important that there are multiple testers, and that some testers are outside the development team. This is to reduce biased or predetermined feedback.

Through testing, we hope to answer the questions:
-Does the system work in a way that is useful and satisfying for users?
-Are there any overlooked errors or illogicalities?
-Is it user friendly?
-Is the interface adequate, understandable and not overly complicated or ambiguous?

Without testing, an unstable and/or unsatisfying system could be delivered to clients.

Feedback

Feedback follows testing. It involves providing testers with means (such as questionnaires) to share what they thought of the system, including downfalls, positivity and suggestions.

Feedback, however, must not only be received from testers, but also from management and other interested stakeholders in order to determine whether or not the physical solution is in accordance with original planning, design and analysis.
It helps address the following concerns:
-Is the system what the client wanted?
-Is the solution what management wanted to be developed?
-Does the system greatly diverge from original planning and design?
-Is the system functioning as it should?
-How can the system be improved?

Improvements

Once testing has been done and feedback received, the system may need to be changed or expanded according to feedback and recommendations.

This means returning to the graphical user interface design and/or coding. Once changes have been made, the new version may have to be tested once again for new feedback. This process may repeat until a desirable and satisfying solution has been reached.

Clerical Level Report

Looking back upon the analyses and the work done on the OmniCal project, it is safe to say that it took no small effort and that a great deal of planning, critical thinking and documentation went into its creation and history.

After sufficient documentation had been drawn up, but before our project could be coded, the stakeholders needed to choose a programming language to code it in.. Looking at the candidate feasibility matrices drawn up by various members, we opted to code in C#, using Microsoft Visual Studio to code. However, the demonstration prototype was developed in Delphi 7.

The stakeholders started by looking at the feasibility analyses for OmniCal, which would prove that this project is more than simply "feasible".
The project fulfilled all the necessary criteria and displayed the potential to become more than just a project, but a hallmark of organization, indispensable in this hectic and sometimes downright confusing society.

Regarding the feasibility of the project, all the various analyses, ranging from cultural to operational feasibility, showed that OmniCal was and is indeed feasible and it will be possible for it to be of benefit to everyone involved.

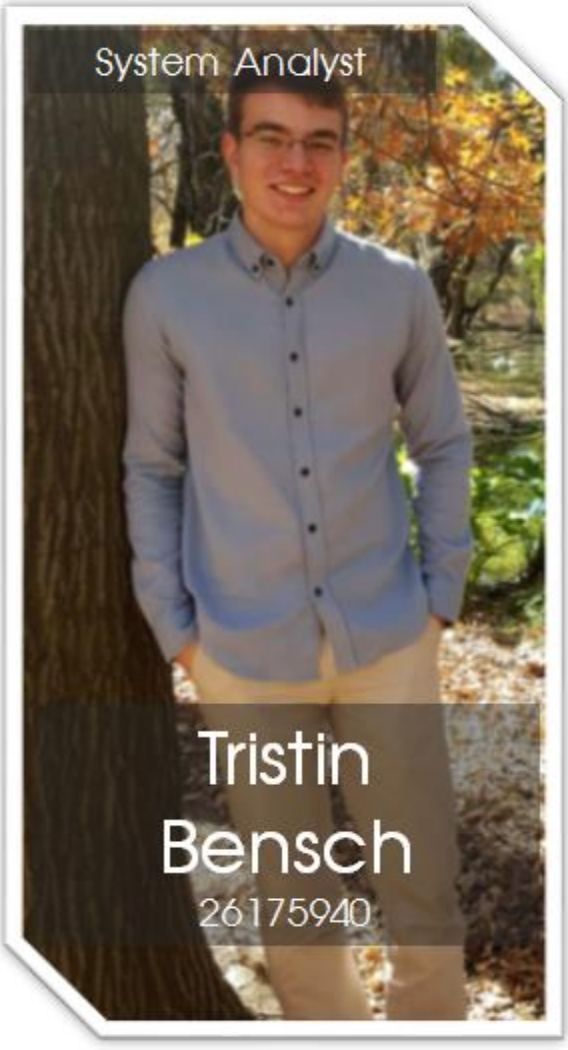
Looking at the cost / benefit analysis, we can clearly see that OmniCal has a very low cost-to-benefit ratio, making it highly cost-effective, and making it feasible in the economic aspect in addition to those of cultural, operational, technical, and legal aspects of feasibility.

While analysing the project's schedule, in terms of feasibility, it may at first seem as if the deadlines and allocated times are too short, making the project seem like an impossible task. However, upon closer inspection, it can be seen that enough time had to be allocated for emergencies, in case of breakdowns, losses of data, or other disastrous and unforseen circumstances, and that the time allocated to each process in the project is reasonable and feasible, allowing for completion, without loss of quality or undue pressure on the team member responsible.

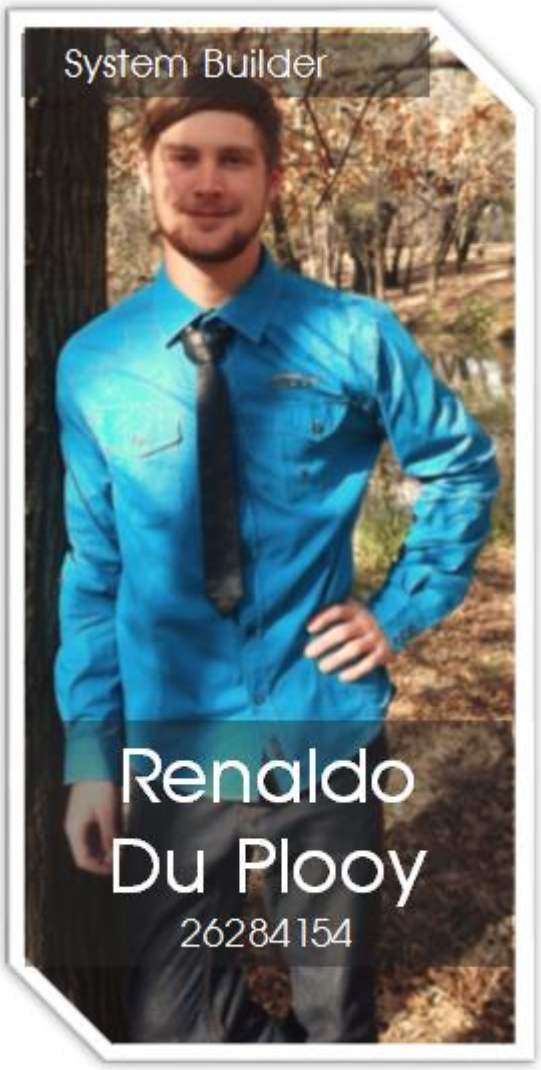
Documenting the entire project, from start to finish, has been a challenge, but was made easier by the weekly assignments (short term goals), and was further simplified by effective use of the textbook.

STAKEHOLDERS GALLERY

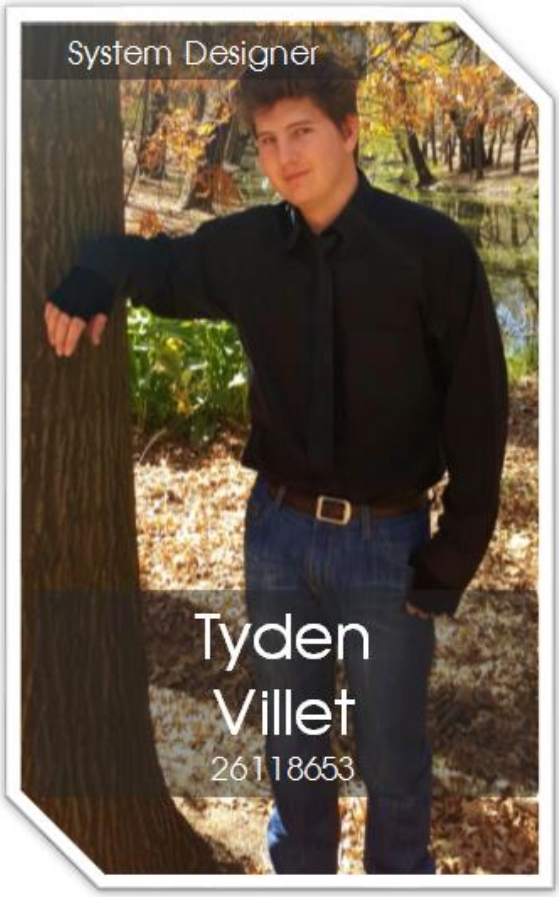
Executive Level



Supervisory Level

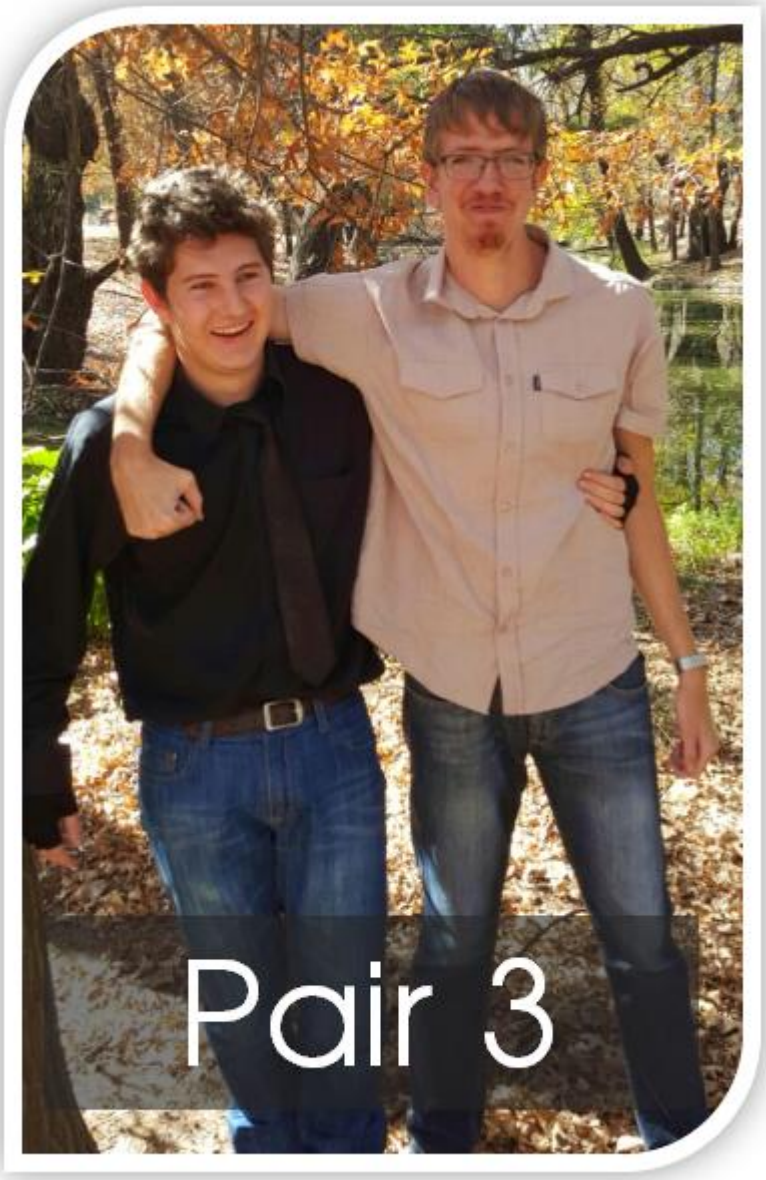


Clerical Level



STAKEHOLDERS GALLERY (Cont.)
MEET THE TEAMS


Pair 2


Pair 3

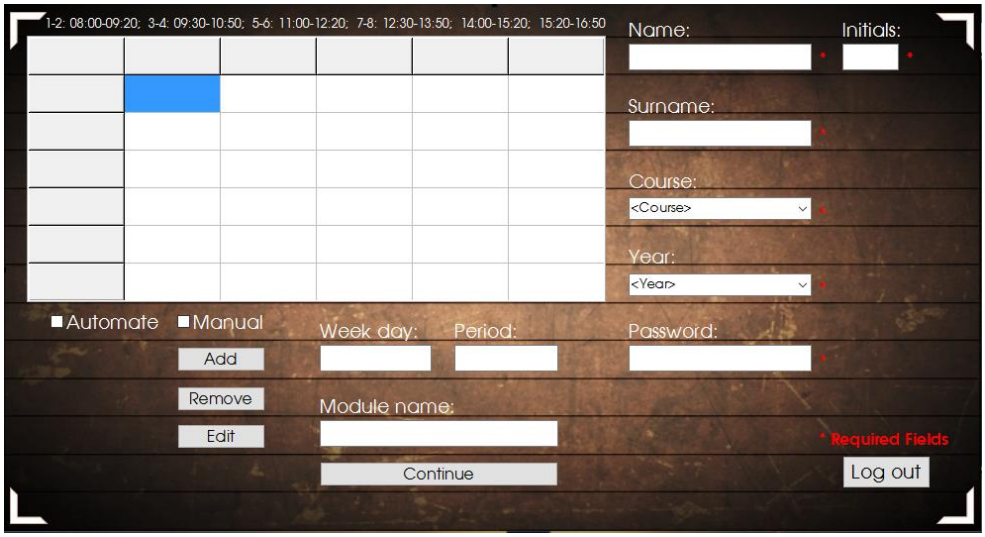
WE GIVE YOU… OMNICAL

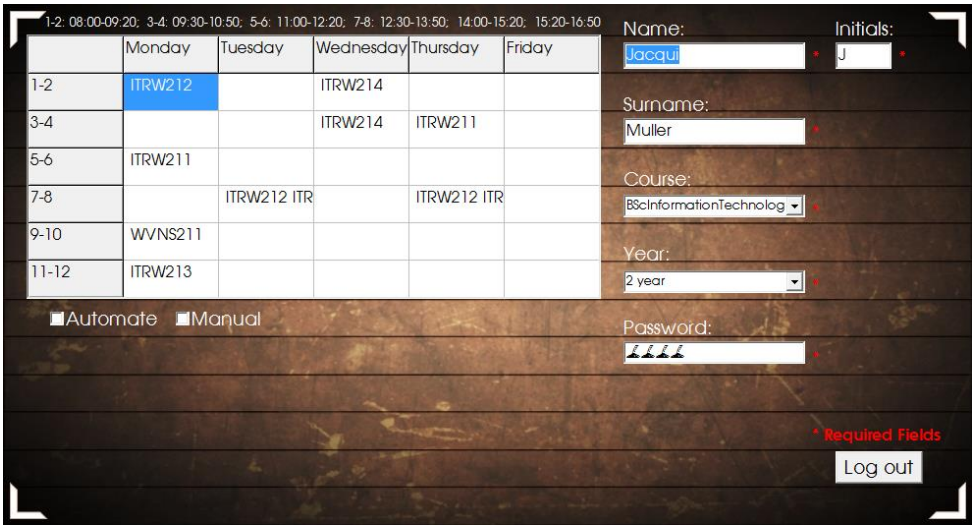

OmniCaL
BSCIT SECOND YEAR FIRST SEMESTER
Calendar 2016
January
February
March
April
May
June
July
August
September
October
November
December
MONDAY:
1-2: ITRW212 12 G01
5-6: ITRW211 3 103
9-10: WVN$211 25 G08
11-12: ITRW213 25 G01
TUESDAY:
1-2: ACC$111-1 12 G01
5-6: ITRW212 3 103
7-8: ITRW212 3 103
WEDNESDAY:
1-2: ITRW214 3-103
3-4: ITRW214 3-103
5-6: ITRW213 3 103
7-8: ITRW213 3 103
THURSDAY:
1-2: ACC$111-1 12 G05
3-4: ITRW211 9B G02
5-6: ITRW212 3 103
7-8: ITRW212 3 103


My TimeTable
OmniCaL
BSCIT SECOND YEAR FIRST SEMESTER
NWU Student
Student Number:
MONDAY:
1-2: ITRW212 12 G01
5-6: ITRW211 3 103
9-10: WVN$211 25 G08
11-12: ITRW213 25 G01
TUESDAY:
1-2: ACC$111-1 12 G01
5-6: ITRW212 3 103
7-8: ITRW212 3 103
WEDNESDAY:
1-2: ITRW214 3-103
3-4: ITRW214 3 103
5-6: ITRW213 3 103
7-8: ITRW213 3 103
THURSDAY:
1-2: ACC$111-1 12 G05
3-4: ITRW211 9B G02
5-6: ITRW212 3 103
7-8: ITRW212 3 103


OmniCaL
BSCIT SECOND YEAR FIRST SEMESTER
NWU Student
Student Number:
26058995
Next >
MONDAY:
1-2: ITRW212 12 G01
5-6: ITRW211 3 103
9-10: WVN$211 25 G08
11-12: ITRW213 25 G01
TUESDAY:
1-2: ACC$111-1 12 G01
5-6: ITRW212 3 103
7-8: ITRW212 3 103
WEDNESDAY:
1-2: ITRW214 3 103
3-4: ITRW214 3 103
5-6: ITRW213 3 103
7-8: ITRW213 3 103
THURSDAY:
1-2: ACC$111-1 12 G05
3-4: ITRW211 9B G02
5-6: ITRW212 3 103
7-8: ITRW212 3 103


tblUser
tblStudent
tblAdmin
tblTimetable
OmniCaL
BSCIT SECOND YEAR FIRST SEMESTER
<User Type>
Sign Up
Log In
Student Number:
Next >
MONDAY:
1-2: ITRW212 12 G01
5-6: ITRW211 3 103
9-10: WVN$211 25 G08
11-12: ITRW213 25 G01
TUESDAY:
1-2: ACC$111-1 12 G01
5-6: ITRW212 3 103
7-8: ITRW212 3 103
WEDNESDAY:
1-2: ITRW214 3 103
3-4: ITRW214 3 103
5-6: ITRW213 3 103
7-8: ITRW213 3 103
THURSDAY:
1-2: ACC$111-1 12 G05
3-4: ITRW211 9B G02
5-6: ITRW212 3 103
7-8: ITRW212 3 103


1-2: 08:00-09:20; 3-4: 09:30-10:50; 5-6: 11:00-12:20; 7-8: 12:30-13:50; 14:00-15:20; 15:20-16:50
Name:
Initials:
Surname:
Course:
<Course>
Year:
<Year>
Automate
Manual
Week day:
Period:
Password:
Add
Remove
Edit
Module name:
Continue
* Required Fields
Log out


1-2: 08:00-09:20; 3-4: 09:30-10:50; 5-6: 11:00-12:20; 7-8: 12:30-13:50; 14:00-15:20; 15:20-16:50
Monday
Tuesday
Wednesday
Thursday
Friday
1-2
ITRW212
ITRW214
3-4
ITRW214
ITRW211
5-6
ITRW211
7-8
ITRW212 ITR
ITRW212 ITR
9-10
WVNS211
11-12
ITRW213
Name:
Jacqui
Initials:
J
Surname:
Muller
Course:
BScInformationTechnolog
Year:
2 year
Automate
Manual
Password:
* Required Fields
Log out


1-2: 08:00-09:20; 3-4: 09:30-10:50; 5-6: 11:00-12:20; 7-8: 12:30-13:50; 14:00-15:20; 15:20-16:50
Monday
Tuesday
Wednesday
Thursday
Friday
1-2
ITRW212 ITR
ITRW214 ITR
3-4
ITRW214 ITR
ITRW211 ITR
5-6
ITRW211 ITR
7-8
ITRW212 ITR
ITRW212 ITR
9-10
WVNS211 W
11-12
ITRW213 ITR
Name:
Kholofelo Poppy
Initials:
KP
Surname:
Mocheke
Course:
BScInformationTechnolog
Year:
2 year
Automate
Manual
Week day:
Tuesday
Period:
1-2
Password:
Add
Remove
Edit
Module name:
ACFS111
Continue
* Required Fields
Log out