

**CSI 473 SOFTWARE DESIGN PROJECT**

**GROUP MEMBERS**

**MASEGO OLEKENG 201802789**

**LAONE MPELA 201802629**

**EDSON PRIDE MUZAMBI 201802792**

**LESEGO ANDREW 201501180**

**MAITUMELO RADIPODI 201804424**

**REFINED SET OF REQUIREMENTS**

**FUNCTIONAL REQUIREMENTS**

* Authentication of a user when he/she tries to log into the system
* Verification email is sent to user whenever he/she registers for the first time on the system
* The system must allow users to reset their password by clicking “i forgot my password” and receiving a link to their verified email address.
* The system must allow the system administrator to add new users, Office Assistants and Doctors
* The system must have an dashboard that is appropriate for their role dashboard role for each user account type
* The system should be able to send notification email to users
* The system must allow online payments
* The system must allow payment through the use if the office assistants
* The system must be secure passwords must be hashed and data encrypted
* The system must have a backup
* The system should backup every 12 hours
* The system should provide statistics to help in decision making
  + **Customers**
* Customers can book to see a doctor.
* Customers can cancel/postpone an appointment with the Doctor.
* Customers can view history of their visits.
* Customers can pay online
  + **Doctors:**
* Doctors use the system during consultation, recording all that used to be recorded on paper. (Diagnosis, prescriptions, referrals, etc.)
* Doctors can make changes to a patient record.
* Doctors can access history of their patients.
  + **Office Assistants can**
* Reject/accept patient bookings
* Print patient’s prescription.
* Capture/record patient’s payment.
* View statistics
* Book for a customer – those who don’t have online access.
* Cancel/postpone a patient’s appointment. The Doctor and the patient will receive an email notification.
  + **System Administrator**
* Is the superuser

**NON FUNCTIONAL REQUIREMENTS**

The requirement imposed on the system .They specify the quality attribute of the system, how the system is supposed to be.

PERFOMANCE:

* Response time: The system should provide acknowledgement in just one second once the user’s information is checked.
* User Interface: The system will be user friendly. Users can view their schedule from anywhere. Patients can make their appointment online from anywhere.
* Capacity: The system needs to support at least 900 users at once therefore a good server is needed and the latest software versions will be used. The software’s will be updated regularly to be able to carry the workload as the capacity increase.
* Availability: Users of the system can access the website throughout the week at any time during the day.

MAINTANBILITY: The system will be easy to be supported and restructured over time.

* Automated data back-up enabled.
* Errors: The system will track/log every mistake.
* Recoverability: The system will be able to resume business functionality upon failure.

PORTABILITY**:** The system software’s should be able to work on different environment and it should be easy for the software running on another platform to be converted to another platform.(e.g. From Windows to Unix)

SCALABILITY:The system should be able to handle extra usage.

RELIABILITY:The system will be able to operate consistently in a user-acceptable manner meeting all the requirements at all the times.

DATA INTEGRITY:Data should be accurate and consistent. They should be no dangers of loss of sensitive data or unauthorised access to data.

**HOW OUR ARCHITECTURE WILL SUPPORT;**

SECURITY

* Implement Role based Access control for our distinct users, Access control list(ACTIVE DIRECTORY)
* Password change prompt after every 90 days

PERFORMANCE

* Implementation of a Distributed system: A system where the computing environment in which various components are spread across multiple computers (or other computing devices) on a network. These devices split up the work, coordinating their efforts to complete the job more efficiently than if a single device had been responsible for the task.
* High performance Server hardware-the hardware should meet minimum requirement to host such a system.
* Our bandwidth should be enough to handle the oncoming traffic and curb system crashes as users increase

SCALABILITY

* Use prepared statements to improve latency
* Optimize CPU and memory usage
* **Constrain concurrent access to limited resources**

DISASTER RECOVERY

* Daily scheduled backup’s every 24 hours

**USE CASE**

The use case diagram could not fit in the page here so we attached it as a pdf file.

|  |  |
| --- | --- |
| Name | Register |
| Participation actors | Patient |
| Entry condition | 1.The use case starts when the customer registers |
| Flow of events | 2.The customer registers a new account in the system  3.The system sends a confirmation link to the customer to confirm registration  4.The system redirects the customer to login  5.The customer logs in  6.Login and registration is successful |
| Exit Condition | 7.The use case terminates when the customer has an account and able to login |
| Exceptions | -Account already exists  -Forgot password |
| Special requirements | None |

**DESCRIPTION FOR EACH OF THE NEW USE CASES IDENTIFIED**

|  |  |
| --- | --- |
| Name | Book an appointment |
| Participation actors | Patient, Office Assistant |
| Entry condition | 1.The use case starts when the user books an appointment |
| Flow of events | 2.The user selects to book an appointment  3.The system processes the appointment(reject/accept/cancel)  4. The system sends a confirmation link to the user to confirm appointment.  4.The user makes a payment  5.Booking appointment successful |
| Exit Condition | 7.The use case terminates when the user completes booking an appointment |
| Exceptions | -Account already exists  -Forgot password  -Appointment schedule conflict |
| Special requirements | None |

|  |  |
| --- | --- |
| Name | Login |
| Participation actors | Doctors, Office Assistant,Patient,Admin |
| Entry condition | 1.The use case starts when the user logs in |
| Flow of events | 2.The user logs in  3.The system prompts the user to enter login credentials  4.The user enters the login credentials  5.The System verifies the login credentials  6.Login successful |
| Exit Condition | 7.The use case terminates when the user logins |
| Exceptions | -forgot password |
| Special requirements | None |

**ACTIVITY DIAGRAM**

The activity diagram could not fit in the page so we attached it as an Argo file.