<UNDECIDABLES>

COSBAS User Manual

Git: `https://github.com/undecidables/Documentation`
GitHub Organisation: `https://github.com/undecidables`

The Client:
Prof Andries Engelbrecht
Head of Department, Computer Science
University Of Pretoria

The Team:
Elzahn Botha *13033922*
Jason Richard Evans *13032608*
Renette Ros *13007557*
Szymon Ziolkowski *12007367*
Tienie Pritchard *12056741*
Vivian Venter *13238435*

August 2015

Contents

1 Introduction

The COSBAS system is a highly secure and modern access control system that uses Biometric data to authorize the individuals request to enter the department and offices. This document will specify how to place the COSBAS system in a working state. The document instructs on the necessary steps to install the hardware needed by our system as well as installing the COSBAS system software on the relevant computers.

2 System Overview

2.1 Overview Description:

The COSBAS (Computer Science Biometric Access System) is a secure system that uses Biometric inputs (such as facial recognition and fingerprint scanning) to unlock and gain access to the department and offices.

2.2 Pinpoint Descriptions:

2.2.1 COSBAS-Client

The COSBAS client is the hardware aspect of the COSBAS system, and should deployed on Raspberry Pi's at the entrances and exits of the departement. It captures biometric data or access codes from the users and sends it to the server for authentication. On successful authentication the door is opened and the user is let into the department.

Currently the COSBAS system supports authentication only through access codes, Facial Recognition and Fingerprint Identification, but the system is highly pluggable so more authentication methods can be added later.

Facial recognition The client takes a photograph that is sent to the server. On the server the image is grayscaled and the most center face extracted for authentication.

Fingerprint Recognition Fingerprint images are close proximity images and hence will always be accurate enough to send it directly to the server for authentication.

2.2.2 Web-Client

The interface for the COSBAS system is a responsive webpage the user (authorized and temporary visitors) may use to request permission for access to the department. They can also book appointments with members of faculty on the system.

2.2.3 Bookings and Appointments

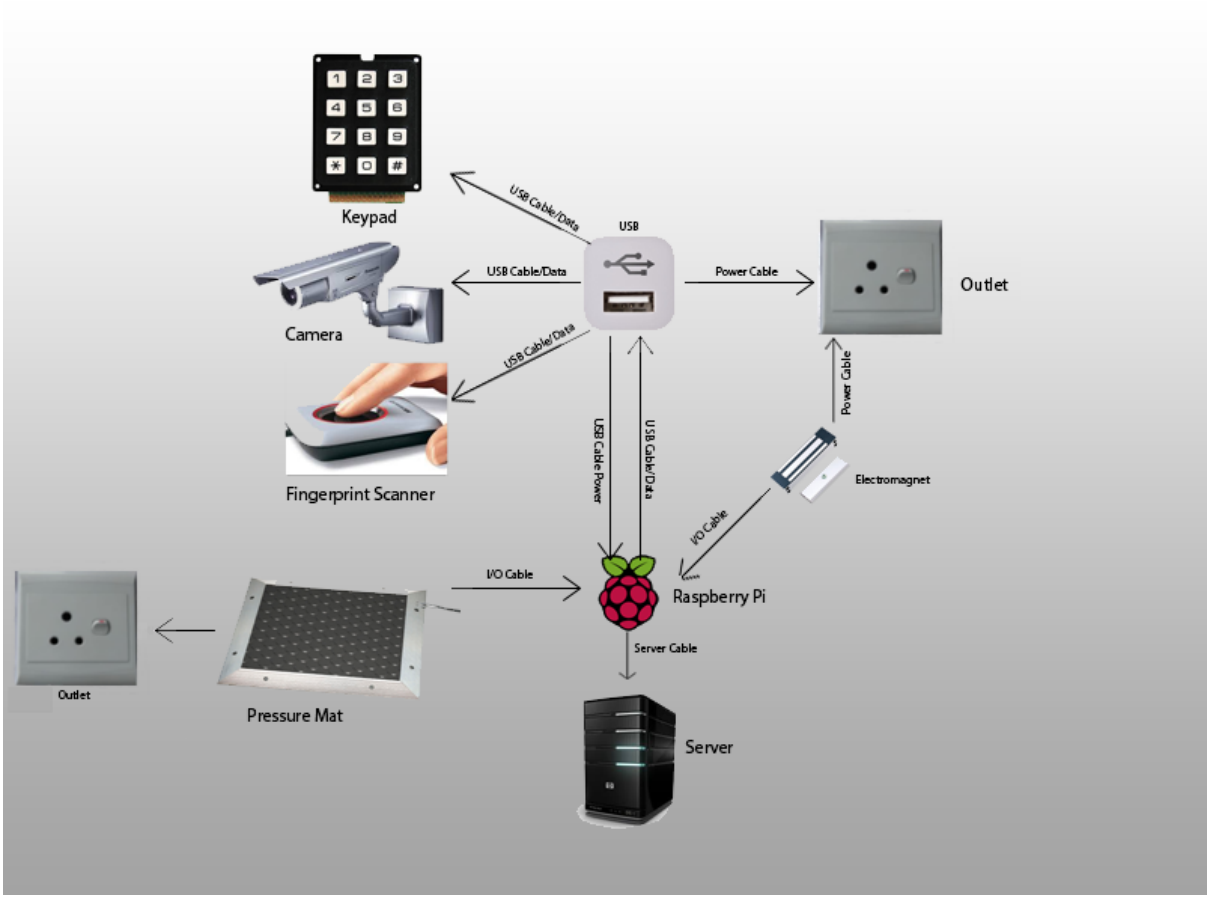
An unauthorized user can book an appointment with an employee enrolled in the system by making use of a Calendar integration feature on the web based interface. Authorized users can then either accept or decline the booking for an appointment of which the person whom made the booking is notified of the status of their booking via an email.

2.2.4 Temporary Access

Once a booking has been accepted by the authorized user of COSBAS, the relevant users will be notified via email containing a link that will expose their temporary access code generated by the COSBAS system. A link is added in the email that the user can click on to cancel the appointment with the associated COSBAS user. In such a case, the temporary access code will be revoked.

3 System Configuration

3.1 Graphical System Configuration Diagram



3.2 Description of the Equipment used as Illustrated on the Diagram

- **Client - Raspberry PI:** Is a very small computer with a very low power consumption. The PI can handle quite a few input/ouput devices via the USB/HDMI/LAN/GPIO ports.
- **Camera:** A device which will capture an image of the user that will want to authenicate via biometrics.
- **Fingerprint scanner:** A device that will capture the fingerprint of the user which will be used for authentication.
- **Keypad:** Will be used by users that will gain access to the building via the keycode.

- **USB Hub:** This device will allow the system to connect more than one device via USB to the client as well as give power to the client.
- **Pressure mat:** This device will allow the system to pick up that the user is ready to be authenticated (usually facial recognition) to gain access to the building.
- **Electromagnet door lock:** Will keep the door locked until the client has succesfully authenticated the user.
- **Server:** Will be used for all the heavy computations such as facial/finger print recognition, etc.

3.3 System Configuration Explained

The system is made up of pluggable authentication devices such as a keypad, camera and fingerprint scanner, as well a client (Raspberry Pi) and a server. The entire authentication process is started a soon as the user steps onto the pressure mat sending the authentication data to the Raspberry Pi for processing before it is sent to the server for authentication. Once authentication is complete on the server, a reply will be sent to the client and the client will act accordingly.

3.4 Communication and Networking

The pressure mat, USB Hub and electromagnet door lock all get their power from the main outlet while the keypad, camera and fingerprint scanner will be getting their power via the USB connection. The Raspberry Pi also connects to the USB Hub both for power and data transfer between the authentication devices and the client. The client communicates with the server via a LAN cable and the pressure mat and electromagnet door lock connects to the client GPIO pins via I/O cables.

Once the user steps onto the pressure mat a signal is sent to the client which gets its data from the authentication devices and processes that data before sending it off to the server if it was valid data. The server the authenticates the person and sends the data back to the client that will then, if the authentication was successful, open the door for the user.

3.5 New Biometric Types

3.5.1 Server Side

Adding a new biometric validator to the system requires source code modification.

1. **Write the validator**
   Each validator should extend the abstract `AccessValidator` class in the `cosbas.biometric.validators` package.
2. **Create a preprocessor**
   If any processing is required on the biometric data sent to the server before a Bioemtric-Data object is created, a new `BiometricPreprocessor` should also be created. If no preprocessing is required, use the existing `NoProcessing` class
3. **Define it as a bean**
   Declare the validator and preprocessor classes as beans by adding the Spring `@Component` annotation right before the class definition.

4. **Register it on the system**
   To register the biometric type on the system add it to the `cosbas.biometric.biometricTypes` enum with its validator and preprocessing class. Eg. `CODE (CodeValidator.class, NoProcessing.clas`
   The type should be uppercase.

3.5.2 Client Side

Adding a new biometric device to the client.

1. **Write the biometric device as a class**
   Write a class which will control the biometric device and implements the `Biometric` interface.
2. **Define is as a bean**
   Declare the class a bean by adding the Spring `@Component` annotation to it.
3. **Register it on the system**

   To register the biometric device, inject the bean into the Factory class and add the data retrieved from the device to the `ArrayList<BiometricData> data`

3.6 The Properties File

3.6.1 Server Side

- `server.port` (port number) The port the server is running on.
- `server.ssl.key-store=keystore.p12` (file name) The SSL keystore file, relative to the project's root directory.
- `server.ssl.key-store-password` (string) The password of the SSL keystore.
- `server.ssl.key-store-type=PKCS12` (string) The type of the SSL keystore
- `server.ssl.key-alias=jetty` (String) The alias of the SSL certificate.
- `ldap.url` (URL) The URL for LDAP authentication.
- `ldap.base` (LDAP base) The base used in LDAP queries.
- `ldap.authpattern` (LDAP dn pattern) The pattern for users that are allowed to log into the system.
- `mongo.address` (IP address) The address of the Mongo database.
- `mongo.port` (port number) The port the Mongo database is running on.
- `mongo.user` (string) The username for the Mongo database.
- `mongo.password` (string) The password for the Mongo database.
- `mongo.database` (string) The name of the Mongo database.
- `codes.newlength` (integer) The length of new access codes.

- `codes.duplicateTries` (integer) Code generation might fail due to a duplicate code in the database. This property id amount of times the server will try to generate a new code before increasing the length. The length increase is not permanent, only for that specific code.
- `google.clientSecret` (string) The client secret needed for using the Google Calendar API.
- `google.clientID` (string) The clientID for using the Google Calendar API.
- `request.pattern.biometric.text` (regex) A Regular Expression. The names of HTTP Request parts containing biometric data should match this expression. If this is changed the clients should also be changed to send the correct name.
- `request.pattern.biometric.group` (integer) The group number from the above mentioned pattern that matches the type of the biometric data.
- `request.pattern.contact.text` (regex) A Regular Expression. The names of HTTP Request parts containing contact details (for registration) should match this. expression. If this is changed the Registration client should also be changed to send the correct name.
- `request.pattern.contact.` (integer) The group number from the above mentioned pattern that matches the type of contact details.
- `faces.certainty` (double, between 0 and 1) The certainty required for positive face validation. It is important that the decimal point should be precede by a
, eg. $0\dot{7}$
- `faces.classifierFile` (resource filename) The xml file that stores the data used to set up the face detection. It is a path relative to the "src/main/resources" folder.
- `faces.imageWidth` (integer, pixels) The width in pixels detected faces are scaled to. This should not be changed after initial system setup as the face recognition algorithm requires images of the same size.
- `faces.imageHeight` (integer, pixels) The height detected faces are scaled to. This should not be changed after initial system setup as the face recognition algorithm requires images of the same size.
- `fingers.threshold` (integer, between 0 and 10) The threshold required for positive fingerprint validation. The default value of 50 is fairly low due to the algorithm used.
- `permissions.newSuper` (userID) If there are no Super Administrators (all privileges) in the database at system startup this user will be saved to the database as one. If an Super Admin already exists, this property will have no effect.
- `scheduling.codesCleanup` (Spring cron expression) The schedule for cleaning old temporary access codes from the database.
- `scheduling.faceTrainer` (Spring cron expression) The schedule for retraining the face recognizer. Actual training will only happen if it detects that it needs training.

3.6.2 Client and Registration

Client Side

- `server.port` (port number) The port the server is running on.
- `server.ssl.key-store=keystore.p12` (file name) The SSL keystore file, relative to the project's root directory.
- `server.ssl.key-store-password` (string) The password of the SSL keystore.
- `server.ssl.key-store-type=PKCS12` (string) The type of the SSL keystore
- `server.ssl.key-alias=jetty` (String) The alias of the SSL certificate.
- `url` (string) The address of the server.
- `port` (port number) Specifies the port on which the Server is running on.
- `map` (string) Specifies the string the server was mapped to in order to validate access request
- `action` (string) Specifies at which side of the door the client will authenticate. Example: if placed outside of the building/door then assign the value of "IN" else if its inside the building assign the value of "OUT"
- `id` (number) Specifies the door number, used for logging.
- `biometric` (string) Specifies the format of the biometric type, must match with the server.

Registration Side

- `server.port` (port number) The port the server is running on.
- `server.ssl.key-store=keystore.p12` (file name) The SSL keystore file, relative to the project's root directory.
- `server.ssl.key-store-password` (string) The password of the SSL keystore.
- `server.ssl.key-store-type=PKCS12` (string) The type of the SSL keystore
- `server.ssl.key-alias=jetty` (String) The alias of the SSL certificate.
- `ldap.url` (URL) The URL for LDAP authentication.
- `ldap.base` (LDAP base) The base used in LDAP queries.
- `url` (string) The address of the server.
- `port` (port number) Specifies the port on which the Server is running on.
- `map` (string) Specifies the string the server was mapped to in order to register a user.
- `biometric` (string) Specifies the format of the biometric type, must match with the server.
- `contact` (string) Specifies the format of the contact type, must match with the server.

4 Installation of the COSBAS System

The COSBAS System consists of two main components:

- The access and appointment server
- The access client

Due to the nature of the system the average user who just wants to use the appointment or access system that is already in place can ignore this section.

The appointment system can be accessed through a normal web browser and gaining access after you have been registered on the system should be as simple as standing in front of the door and entering an access code or activating the biometric devices.

4.1 Obtaining the Software

The system's Java source code as well as all related documentation can be found on the Undecidables GitHub organisation at `https://github.com/undecidables`

The important repositories in this organisation:

- **Documentation:** The documentation repository is home to the project's Wiki and also contains the Functional Requirements, Architectural Requirements and User Manual.
- **COSBAS-Server:** This repository contains the server component of the project. It consists of Java sourcecode, Thymeleaf view templates, a Gradle build file and a few configuration files.
- **COSBAS-Client:** This program is the client application to request access through the biometric system. It is also written in Java and uses the Gradle build system.

The following two repositories are less important to the end user, but might give some insight into the beginning of the system's development:

- **Research:** This repository was created as a central location for the reasearch conducted at the beginning of the project, especially reasearch about hardware, technologies and frameworks.
- **Tenders:** This repository contains the tender documents the team created for the original COS301 project proposals. It is not important to a user of the COSBAS system.

4.2 Installing the Server Component

4.2.1 Configuration

Common system properties such as the server port, the LDAP address and the mongoDB address can be set in the application.properties file located in 'src/main/resources'. The so-called 'secret' file needed to use the Google Calendar API should also be placed in this location.

4.2.2 Building

We use the Gradle build system to manage dependencies:

- On a system that has Gradle 2.3 installed simply run 'gradle build' to created an executable jar and use 'gradle run' to execute it.
- On a system that does not have at least Gradle 2.3 installed, the gradle wrapper (that is on the repository) can be used. Simply use 'gradlew' instead of 'gradle' when you are in the projects root directory. (On a linux system maybe './gradlew'). This wrapper will then download the correct version of Gradle and use it to build the project.
- gradle and gradlew might require super user or administrator privileges.

4.3 Installing the Client Component

The client component should ideally be deployed on a small computer like a Raspberry Pi located at a door or some other access channel. The hardware configuration for the client was discussed in section 3 in this document.

4.3.1 Configuration

Similarly to the structure used in the server component common application properties (eg. where the door it is located at and the server's address) can be set in the config.properties file located in src/main/resources.

4.3.2 Building

The client also uses the Gradle build system so it can be built and executed in the same way as the server (see section 4.2.2).

Due to the minimal resources of the Raspberry Pi's and to save some time it is recommended that the client is built and set up on one Pi and then cloned to the others. Small configuration changes can be made to each Pi at a later stage.

5 Getting Started

5.1 Getting Access to the System

To gain access to the COSBAS System through the web client you need the following:

- **Username** - This username is the same as the username needed to login to the CS Website. Usually it is the employee number of the staff member.
- **Password** - This is the password that you use to login to the CS Website.

5.2 Register on the System

Authentication for the appointments system is through the departments existing LDAP system, no external users can access the system unless they are added to LDAP first.

Note: Only staff members or frequent recognised members will be able to get access to the system by means of the CS username and password. If you are not such a member then you can view the website as a guest and request appointments.

5.3 Change of Login Details

The COSBAS System is not be able to change your username or password. COSBAS login details are the same as your normal Computer Science login details that is managed by the department of Computer Science

5.4 General Walkthrough of the System

5.4.1 COSBAS-Client

As mentioned in section 2.2.1, the client consists of the hardware, which is the Camera, Fingerprint Scanner, Raspberry Pi, Keypad and Pressure Mat. The client captures the biometric data and sends it to the server for authentication.

Walkthrough per Biometric/Authentication Method,

- **Facial Recognition** - Stand on the pressure pad to initialize the camera and take a photo. After an photo has been taken. The image is sent to the server for face detection and authentication. If the authentication is successful, the user can gain access to the department.
- **Fingerprint Scanning** - Place a finger on the fingerprint scanner.
  Either one of the following fingers may be used,
  - Left Thumb
  - Left Index Finger
  - Right Thumb
  - Right Index Finger

  The fingerprint scan will be authenticated against current stored copies of the user's fingerprints. The user will gain access if the authentication has been successful.
- **Authentication Key** - Enter the authentication key on the keypad.

  - For **registered users** such as the staff members this will be the dedicated authentication key you will be provided with once registration for the COSBAS System has been done.

  - For **temporary/guest users** this will be the key that was provided to you via email after the appointment has been approved by the particular staff member.

5.4.2 Web-Client

- **Registered User**
  - Login with your COSBAS Login Details (see section 5.1).

- If you have a gmail account and would like to link your Google Calendar to the appointment system, grant permission to Google to get access to your Calendar (Upon logging in, you will be automatically prompted to link your Google account to your COSBAS account).
- Go to the My Appointments page to approve or decline appointments.
- Go to the Request Appointment page to make a booking by using the online form.
- Go to the Check Appointment page to check the status of an appointment.
- Go to the Cancel Appointment to cancel an appointment.

- **Guest User**
  - No login will be needed.
  - Go to the Request Appointment page to make a booking by using the online form.
  - Go to the Check Appointment page to check the status of an appointment.
  - Go to the Cancel Appointment to cancel an appointment.

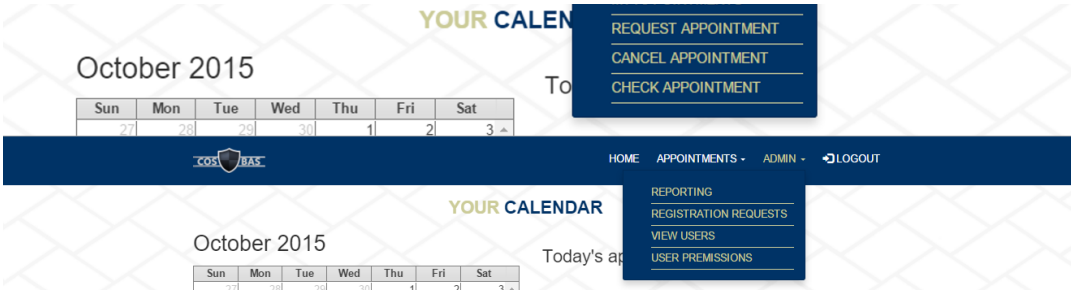
5.5 Exit the System

To exit the system depends on the type of user you are,

- if you are a **registered/admin user** you can simply click on the logout button to exit the system.
- if you are a **guest user** you can exit the system by simply closing the browser.

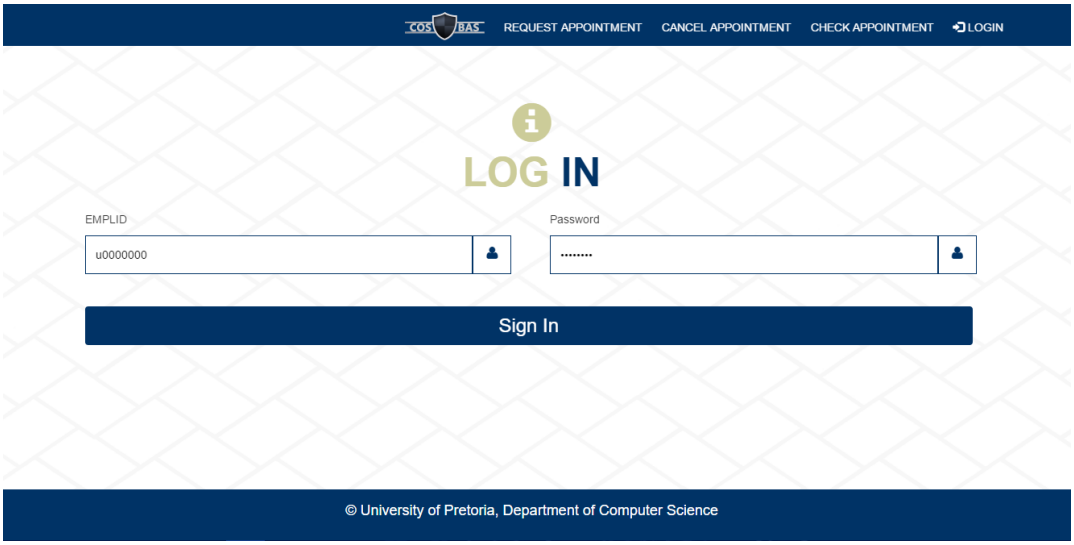
6 Using the System

6.1 Using the Appointment System

- Navigation
  - 

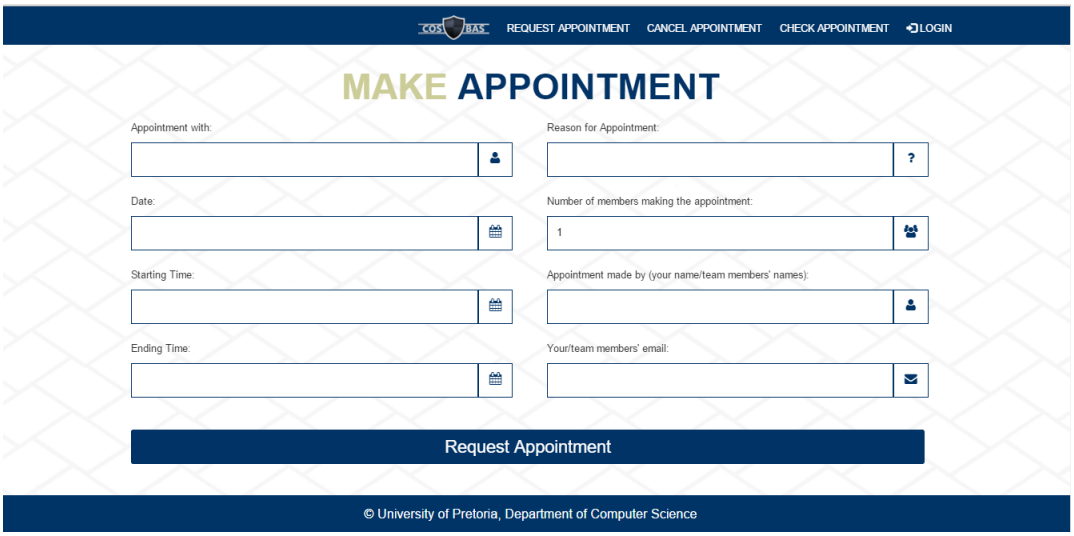
  - The navigation is simple to use. It consists of 5 pages which a user can navigate to and use. These pages are:
    * My Appointments
      · This page can be used to view pending appointments and approve or deny them.
    * Request Appointment
      · A appointment requester will use this page to request an appointment with a lecturer.
    * Cancel Appointment

        - An appointment requester or lecturer can use this page to cancel an appointment.
    - Check Appointment
        - A appointment requester will use this page to check if their appointment was approved by a given lecturer.
    - Reporting
        - An authorized user can generate reports about the system such as appointments made, exit and entrance times etc.
    - Reqistration Requests
        - An authorized user can approve or deny users' requests to be registered on the system
    - Registered Users
        - An authorized user can remove other users from the system. All other registered users can see who is on the system.
    - User Permissions
        - An authorized user can change other users' permissions as well as add and remove permissions

- Login
  - 

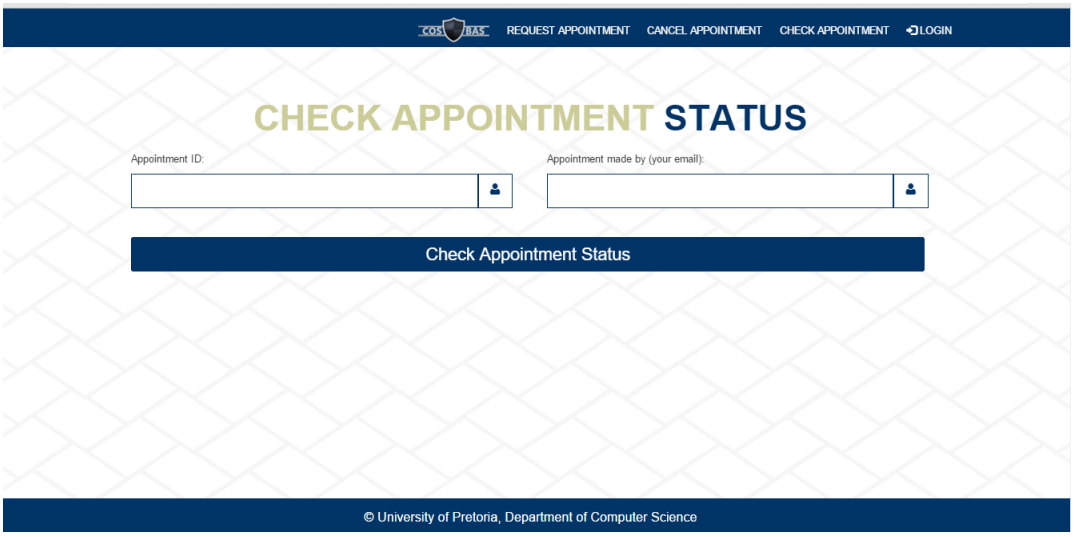
  - This page is used to provide authentication to the system. The user must enter their EMPLID and password, which is authenticated with the CS department's LDAP system.

- Request Appointment



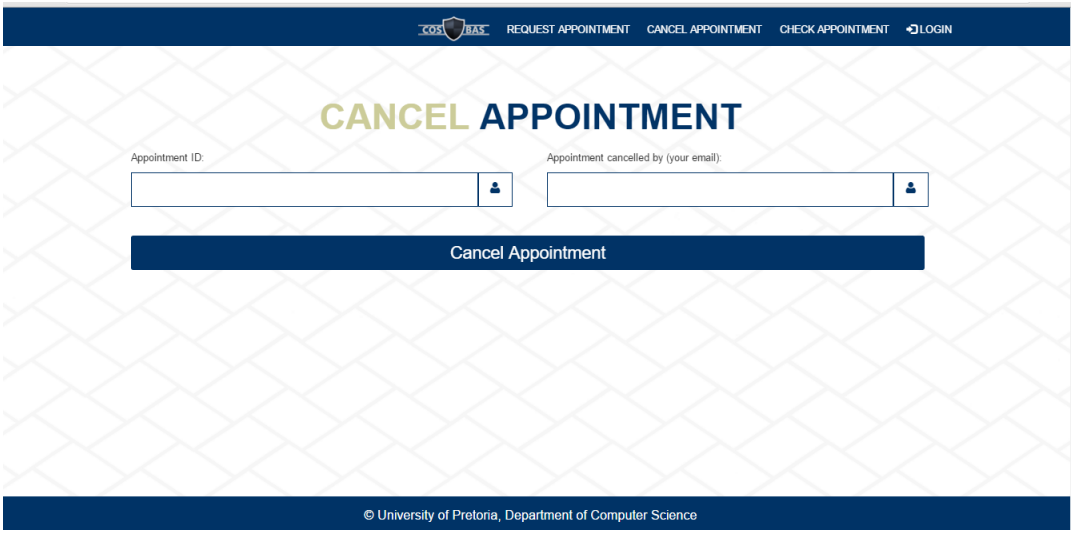
  - A student can request an appointment by selecting the intended staff member from a dropdown list.
  - They can then select the date and time of the meeting.
  - They can delegate how many people will be attending the meeting.
  - They must then give the names of all the people attending the meeting.
  - They must then specify why they want the appointment
  - They must then specify how long they want the appointment to last.

- Check Appointment



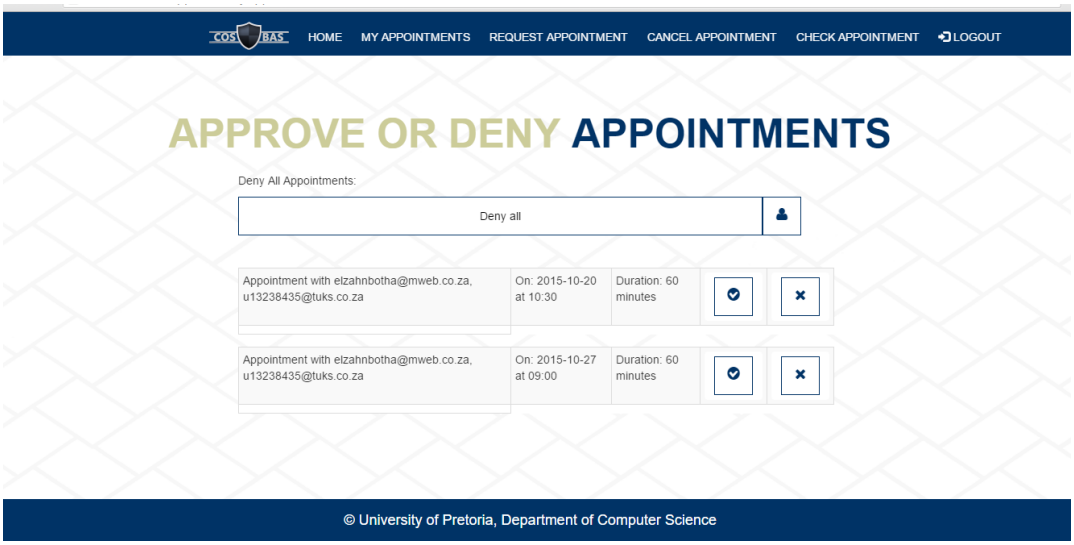
  - Users can check the status of an appointment by entering the unique appointment ID which will be e-mailed to the person who made the appointment.
  - They must also provide the email of the person who made the appointment.

- Cancel Appointment



  - To cancel an appointment, the user must enter the appointment ID which was e-mailed to the person who requested the appointment.
  - They must also enter the email of the person who requested the appointment.

- Approve or Deny Appointments

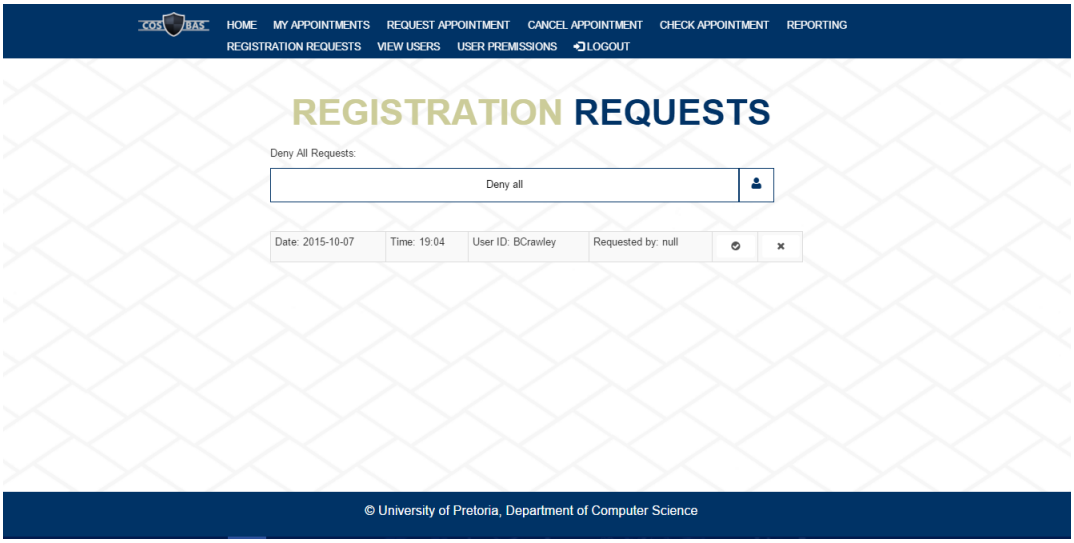


  - To approve or deny an appointment, go to the My Appointments page after having logged in. The user clicks on the appropriate appointment's approve or deny button.
  - Once approved or denied both parties are notified, via email, of the appointment's status change.

- Reporting

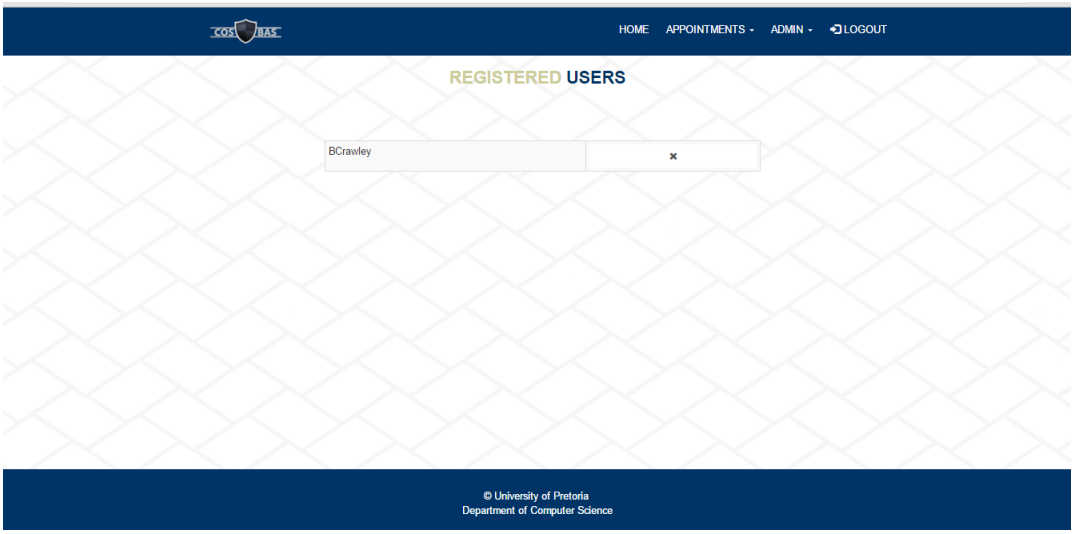
  - In order to create and view reports user must be logged in and have the necessary permissions otherwise an Access Denied error will be displayed.
  - Once the user has logged in and navigated to the Create Reports page they will have to select the type of report they want and in what format.
  - Based on the users selection of type of report, certain inputs will be displayed which the user will have to fill in.
  - User may then press on the Create Report button after filling in all the required inputs.
  - A download dialog will then appear allowing the user to save to report to their local disk.

- Registration Requests



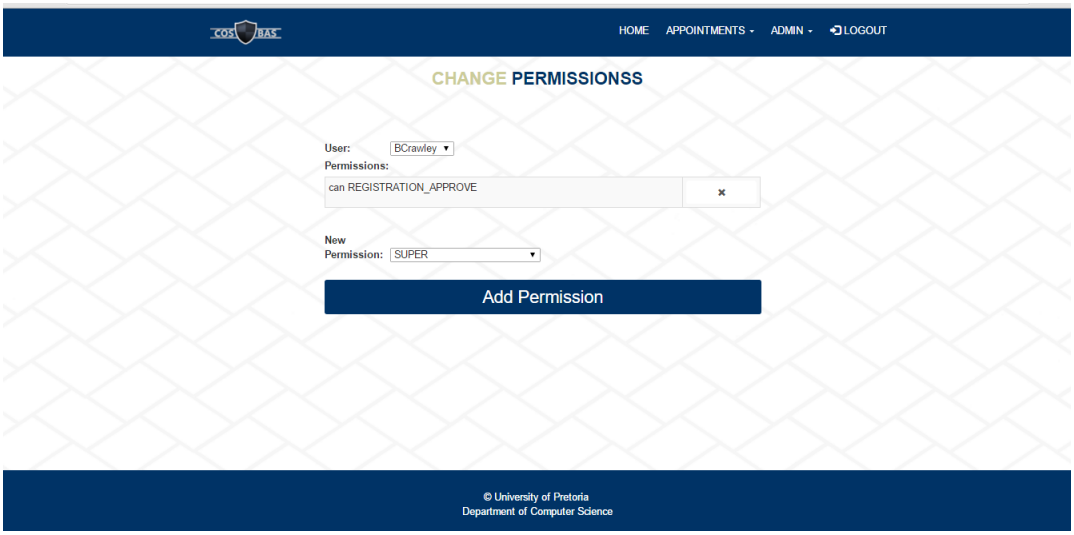
  - To approve or deny a registration request, go to the Registration Requests page after having logged in. If the user has the correct permissions the user can click on the appropriate request's approve or deny button.
  - If the user doesn't have the correct permissions they will be denied access to the page.

- Registered Users



  - To remove a user from the system, go to the Registered Users page after having logged in. If the user has the correct permissions the user can then click on the appropriate user's remove button.
  - If the user doesn't have the correct permissions the user will not see the remove buttons.

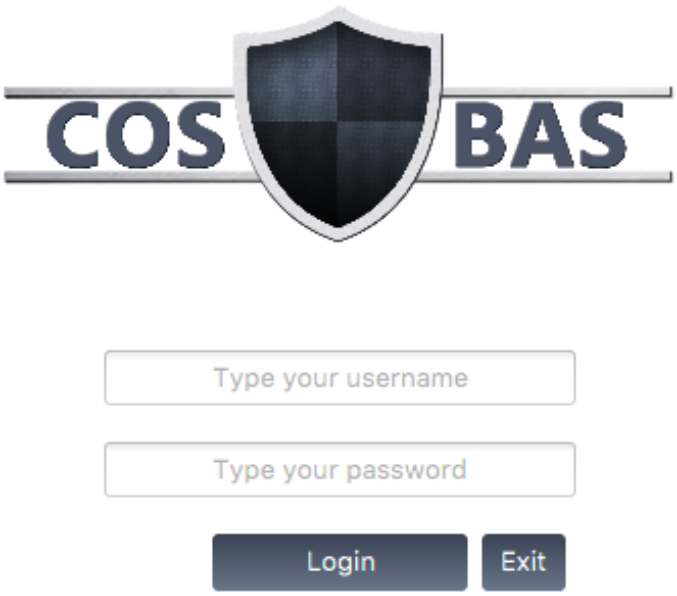
- User Permissions



  - To change a user's permissions or add or remove permissions, go to the User Permissions page after having logged in. If the user has the correct permissions the user can then change a user's permissions.
  - First choose a user from the top drop down.
  - To remove a permission click on the appropriate permission's remove button.
  - To add a permission, choose the correct permission from the drop down and click on the add Permission button.
  - If the user doesn't have the correct permissions the user will be denied access to the page.

6.2 Using the Registration Application

- Logging In.

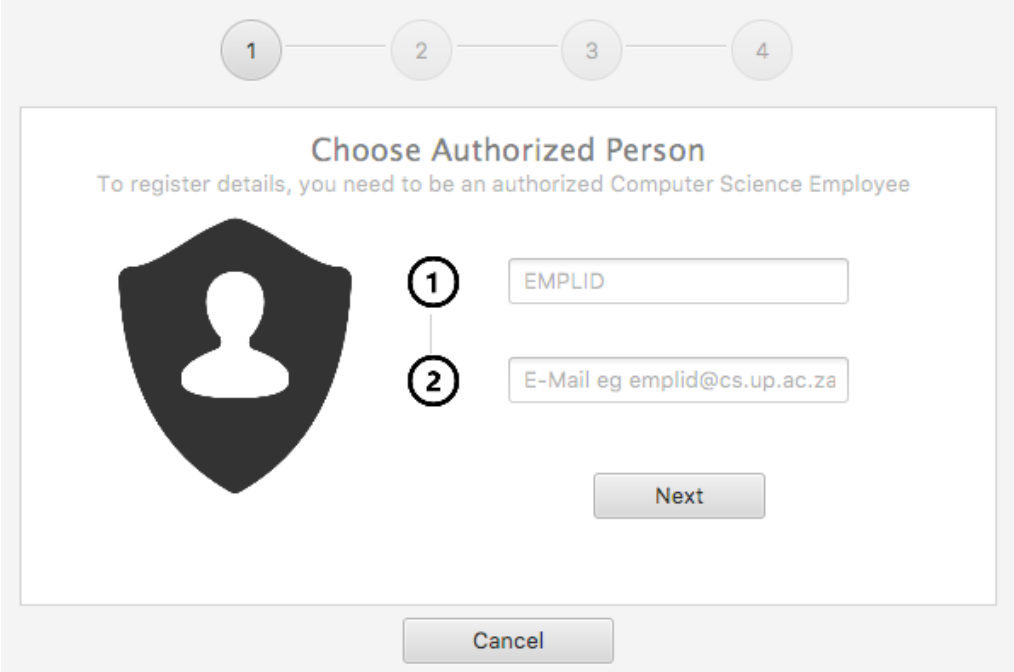


  - This page is used to provide authentication to the registration system. The authorised user should enter their EMPLID and password, which is then authenticated with the departments LDAP system, in order to gain access.

- Landing Page.

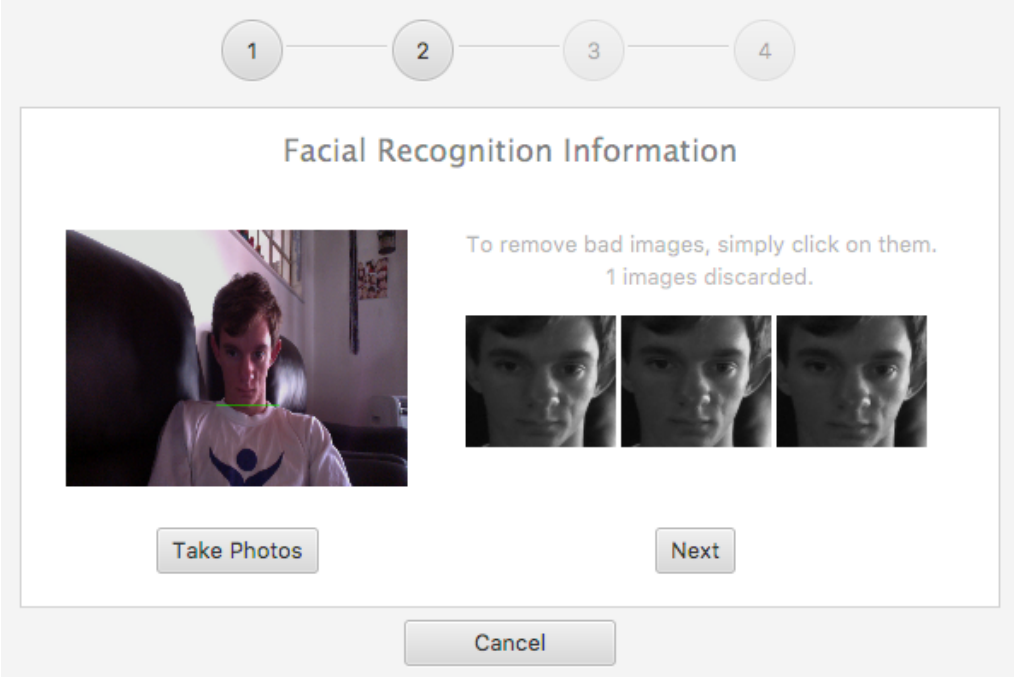
  - When a user has successfully authenticated and logged into the COSBAS registration application, the user will be redirected to the landing page.
  - The user will be able to start a new registration process from this page or simply log out of the application.

- Registration Process.
  - Personal information.



*
  * On this step of the registration process, the user is asked to supply his EMPLID, which is authenticated with LDAP, and their preffered email address through which communication will occur.

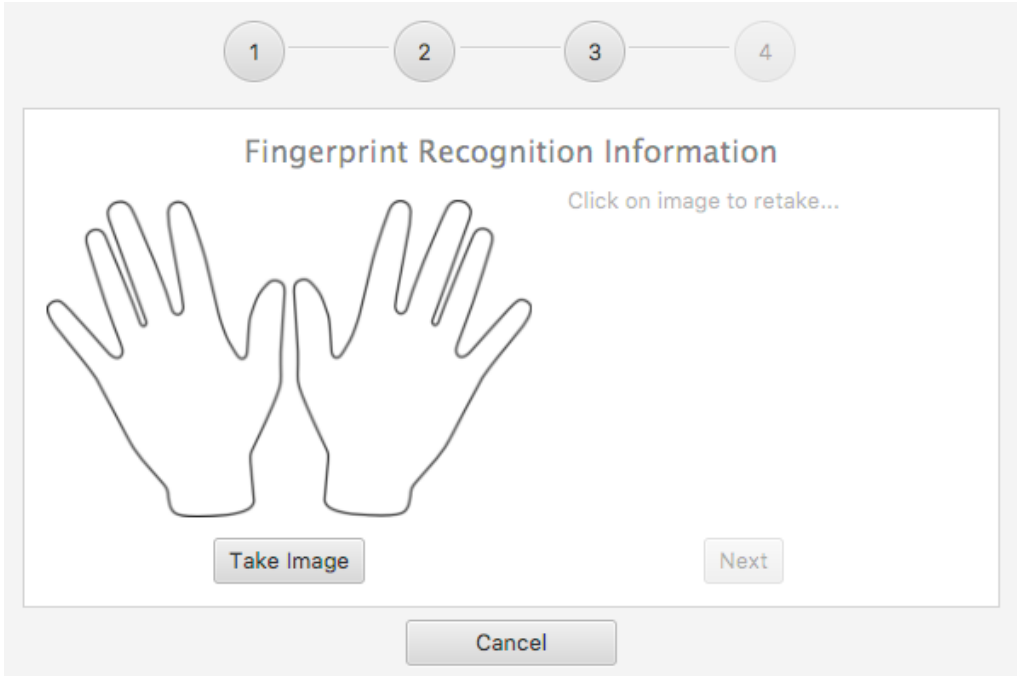
– Facial Recognition Data.



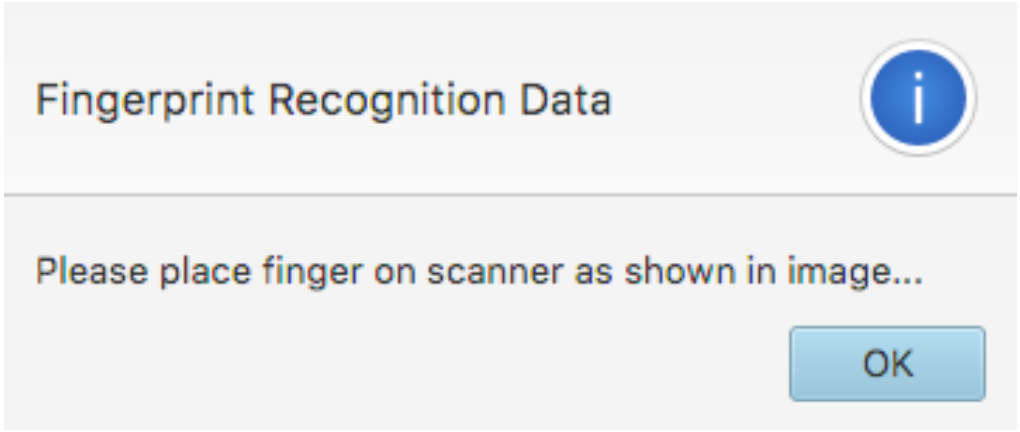
*
  * On this step of the registration process, the user will be asked to take images of their face that will be used to authenticate them via Facial recognition.
  * Once the user selects the take image button, a prompt will be given to ask if they wear glasses or not.

*
    * Once the user has taken images, and finds that one image is not according to his/her liking, then they can click on the image to discard it and replace it with another.

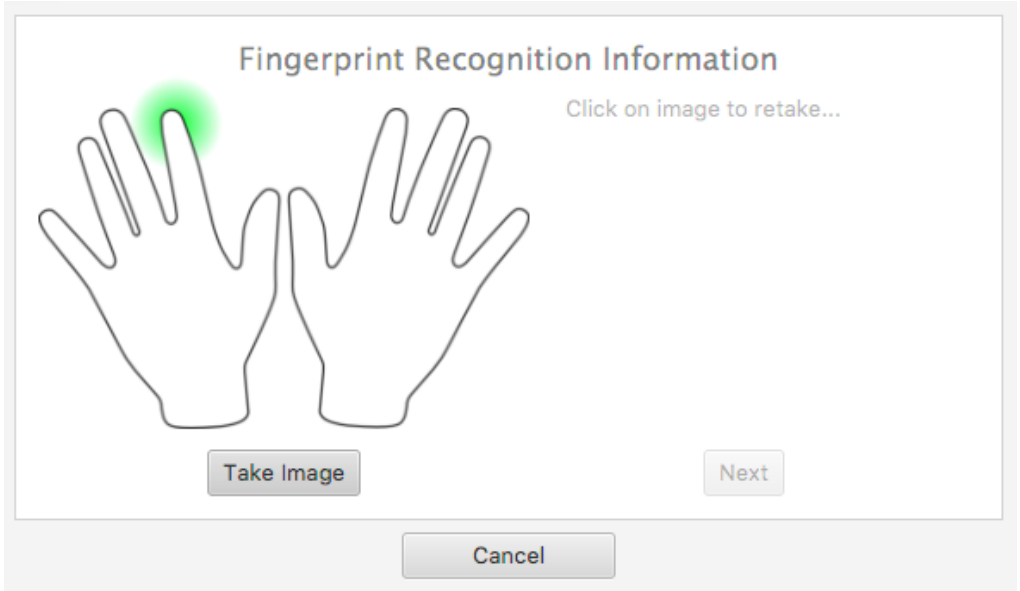
– Fingerprint Recognition Data.



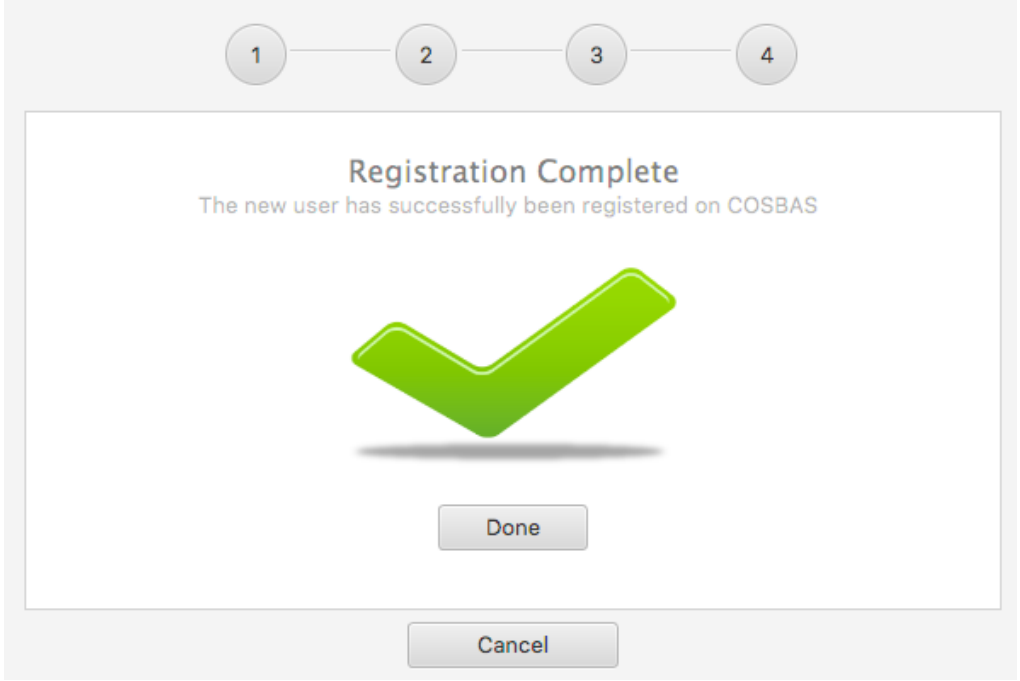
*
    * This step will allow the user to capture fingerprint data needed for fingerprint recognition.
    * When taking images for fingerprint recognition, the user will be promted with a request to place a finger on the scanner.



* The user will also be shown on the UI which finger they will need to place on the finger.



– Registration Completion.



* This step will display if the registration is complete or not. If something wrong happened during registration, then a message will be shown to inform the user to retry the registration.

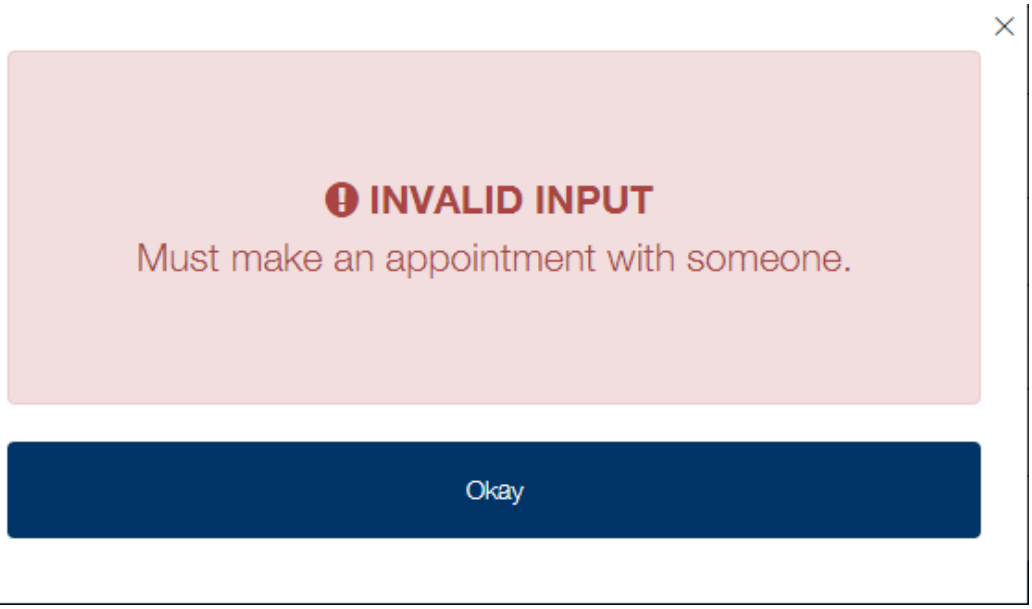
6.3 Using the Client

- Authentication
  - User will have to stand on the pressure mat in order to start up the authentication process.
  - The user will then be asked to look at the camera and capture a few images.
  - After the images have been captured the user will be asked to place their finger on the scanner in order to capture the finger print of the user.
  - The request will be sent to the server for validation, once a response has been received, the door will open or remain closed depending on the response from the server.

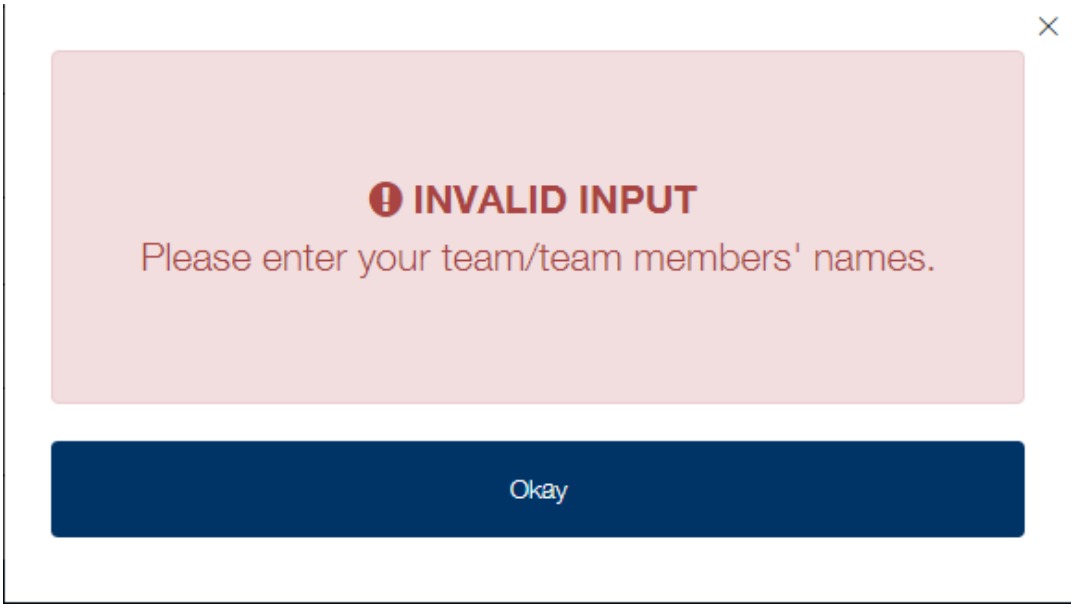
7 Troubleshooting

7.1 Errors while using the Appointment System

- Login
  - Login Credential errors can occur when either an incorrect Employee ID and/or password is entered.
  - How to fix the error: Re-enter the correct password and/or Employee ID
- Request Appointment

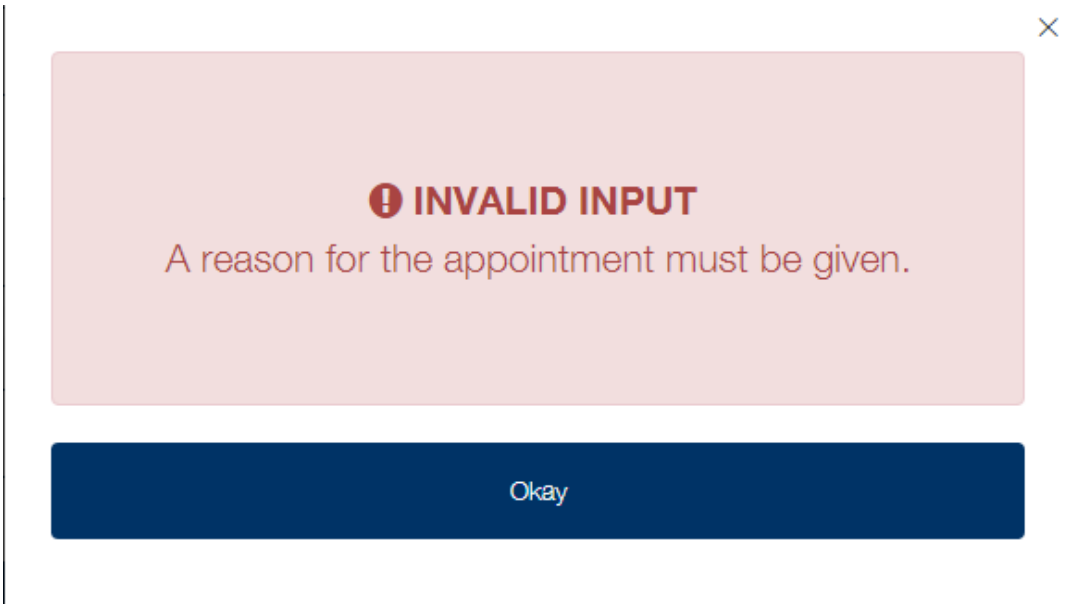


  -
  - Error: "Must make an appointment with someone"
  - How to fix: Choose a staff member from the drop down.
  - Error: "The minimum of people making an appointment is one"
  - How to fix: The minimum people that can make an appointment of one. Any number less than one will give this error. To fix it just input a positive number.



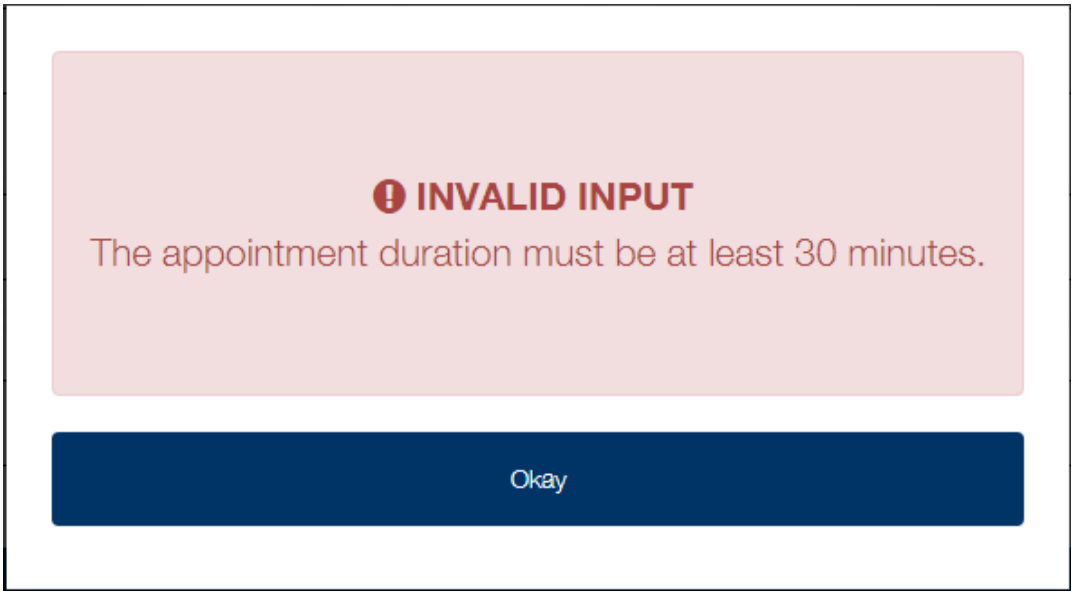
– ”All member names must be given.”

– How to fix: For each member in the group you must provide their name. No input can be left empty.

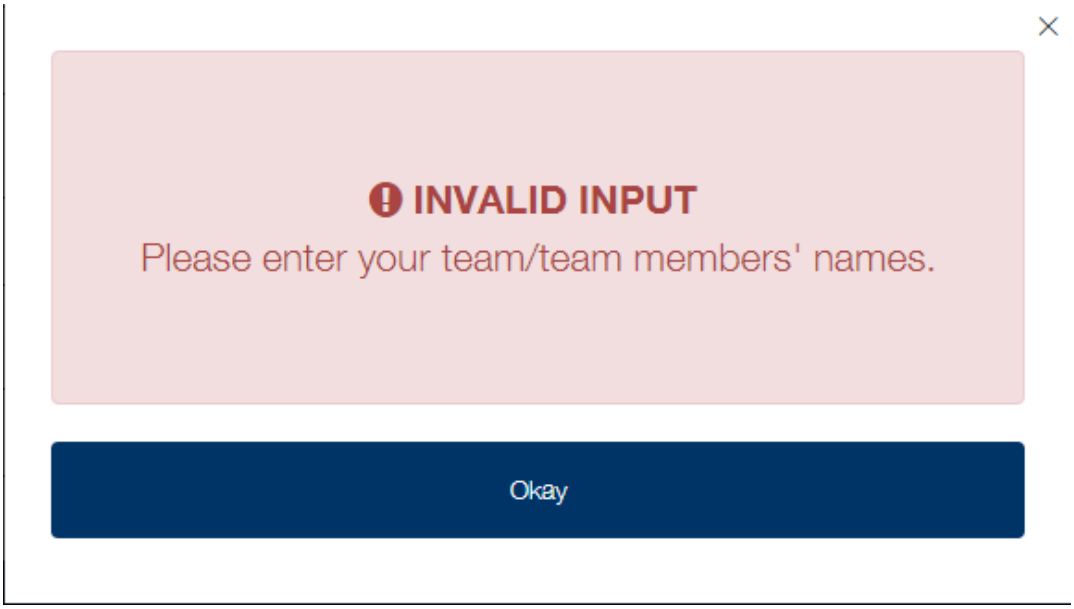


– Error: ”A reason for the appointment must be given.”

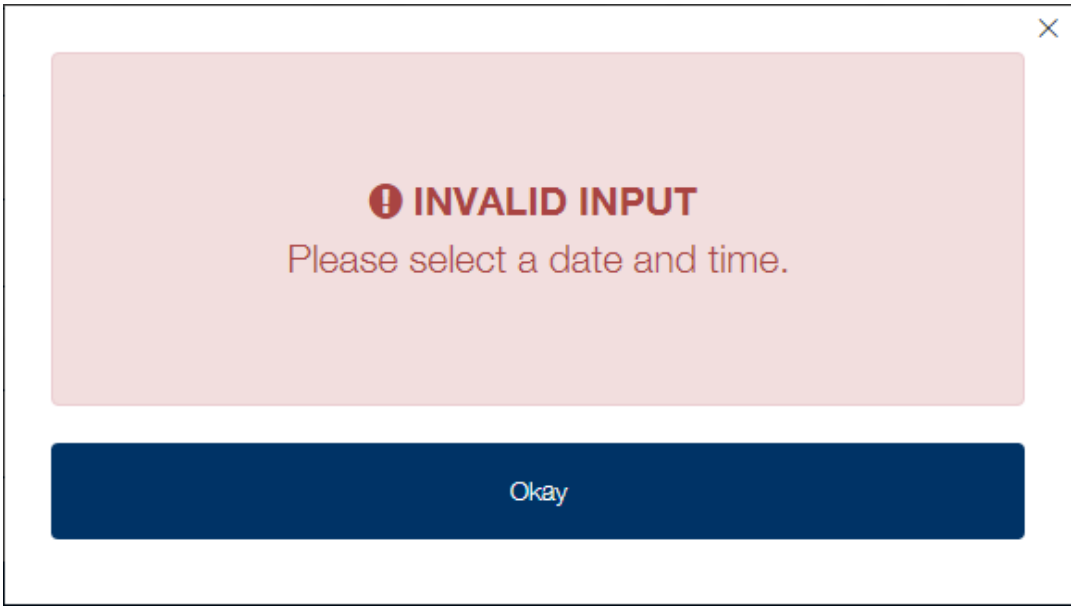
– How to fix: A reason for the appointment must be given.



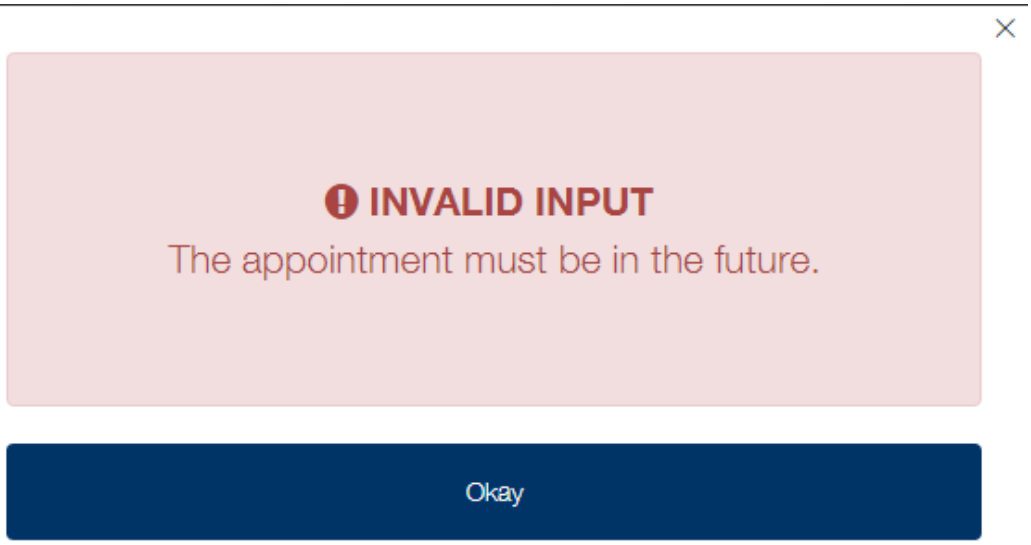
– Error: "Your appointment duration must be at least 30 minutes."
– How to fix: The duration of the requested appointment may not be less than a 30 minute duration.



– "All member emails must be given."
– How to fix: For each member in the group you must provide their email. No input can be left empty.

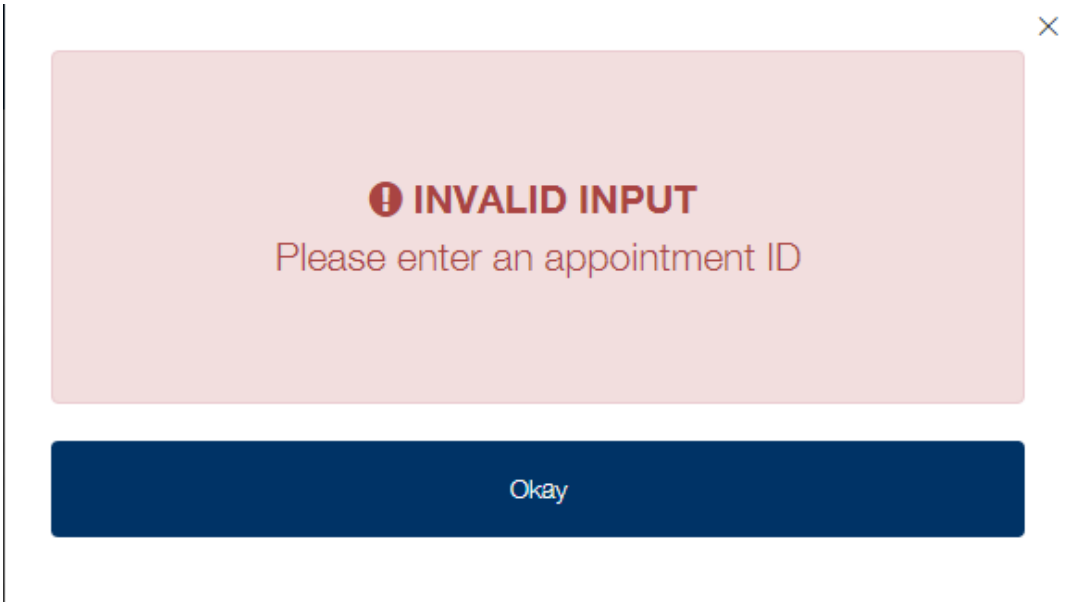


  -
  - Error: "Please select a date and time"
  - How to fix: Select a date and time for the appointment to take place

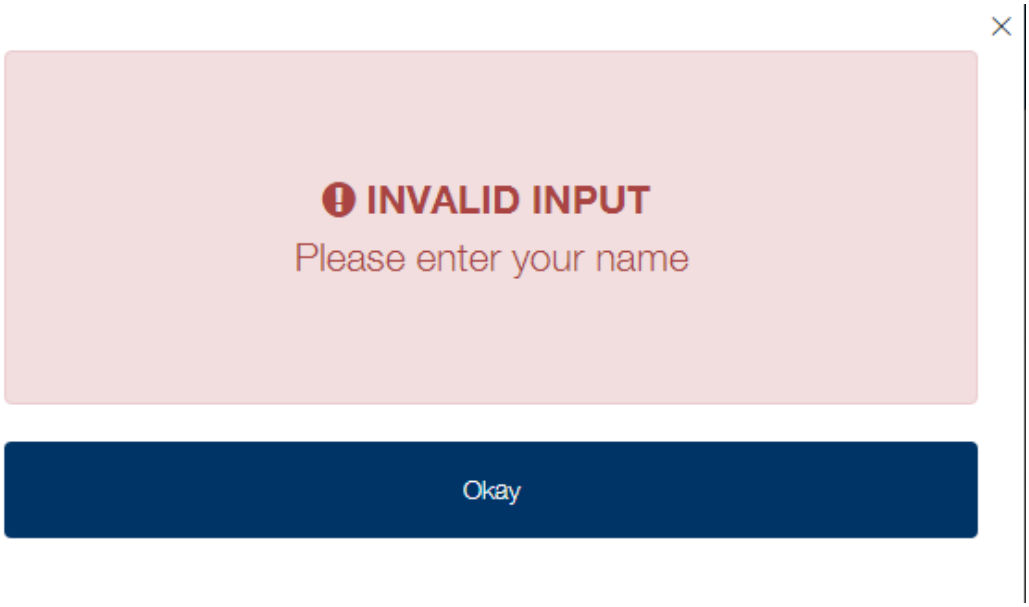


  -
  - Error: "The appointments must be in the future."
  - How to fix: Choose a date and time that is still to come. The tie can not have already past or be the current time.

- Check Appointment

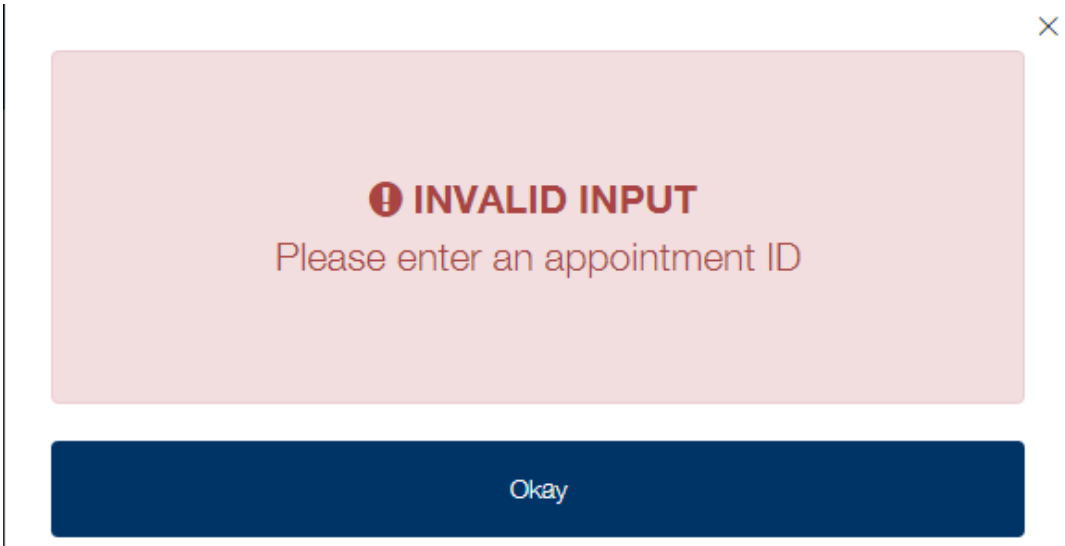


  - Error: "A valid appointment ID has to be entered"
  - How to fix: To fix this error you need to enter a valid appointment ID to be checked.



  - Error: "You must enter who it is that wants to check the appointment's status"
  - How to fix: This input can not be left empty as the person requesting to check the appointment has to be one of the people that's involved in the appointment.

- Cancel Appointment



– Error: "A valid appointment ID has to be entered"
– How to fix: To fix this error you need to enter a valid appointment ID to be checked.

– Error: "You must enter who it is that wants to cancel the appointment"
– How to fix: This input can not be left empty as the person requesting to cancel the appointment has to be one of the people that's involved in the appointment.