

SmartChip – keep it simple



**keep it
simple.**

Android app & Desktop app

Instructor and mentor: Dr. Ilan Kirsh

Students:

David Tsap, ID 307894857

Sigal Bindman, ID 305280968

Tal Mizrahi, ID 201050721

Amit Dror, ID 303042667

SmartChip – keep it simple

Overview:

Smart Chip is an android application that makes execution operations on your mobile more efficient.

Through the app user can purchase NFC chips and encode them to execute various actions.

NFC (Near-field communication) is a set of communication protocols that enable two electronic devices, one of which is usually a portable device such as a smartphone, to establish communication by bringing them within 4 cm of each other.

User account holders can manage their chips by:

- Change the execution action of a chip.
- Edit chip name.
- Delete existing chip.
- Encode new chips (supported only in android app).

In addition, we have developed a desktop app throw it the user can manage his chips account in a form of editing existing chips name and editing executable action.

SmartChip – keep it simple

Smart chip support 2 types of chips:

1. Personal chips – executable action will be activated only in user personal mobile device.
these chips can be edited in both apps (android and desktop app).
2. Global chips – executable action will be activated in any android mobile device (that NFC is turned on).
Once chip has been encoded, it cannot be encoded again; which means user can't re-edit these chips.

Smart chip product can be used by a variety of users: from elders that have technological difficulties, to every one of us that wants to save time and make everyday actions in a more efficient and less time-consuming way.

In the following sections we'll cover the functionality and architecture in details of boat apps.

SmartChip – keep it simple

Android app:

Application functionality:

Main Startup Activity:

- Button **Register** for a new user – will transfer the user to **'RegisterActivity'**.
- Button **LogIn** for app user – will transfer the user to **'LoginActivity'**.
- Button **Exit** – will close the app.



SmartChip – keep it simple

Register Activity:

- Textbox to enter the **User Name**.
- Textbox to enter the **Password** – includes validation of the password, a proper message will be displayed. (shorter than 8 chars)
- Textbox to enter the **Email Address** – includes validation of the address, a proper message will be displayed. (not valid mail)
- Textbox to enter the **Mobile Phone** – includes validation of the phone, a proper message will be displayed. (10 digits)
- Button to **Select Picture** for the user's profile picture – includes validation of the picture, a proper message will be displayed. (JPEG or PNG and less than 500KB)
If no picture were selected, a default user profile picture will be displayed.
- Buttons for **Info** about each data that the user needs to insert:
 - **User name info:** "Username must contain at least 3 characters".
 - **Password info:** "Password must contain at least 8 characters".
 - **Email address info:** "You must enter a valid email with @ that is not already registered to the app".
 - **Mobile phone info:** " Phone must be formatted as: xxx – xxxxxxxx".
 - **Picture info:** " Image must be PNG / JPEG".
- Button to **Sign In** the new user with the entered credentials (will be disabled if one or more of the credentials isn't valid).

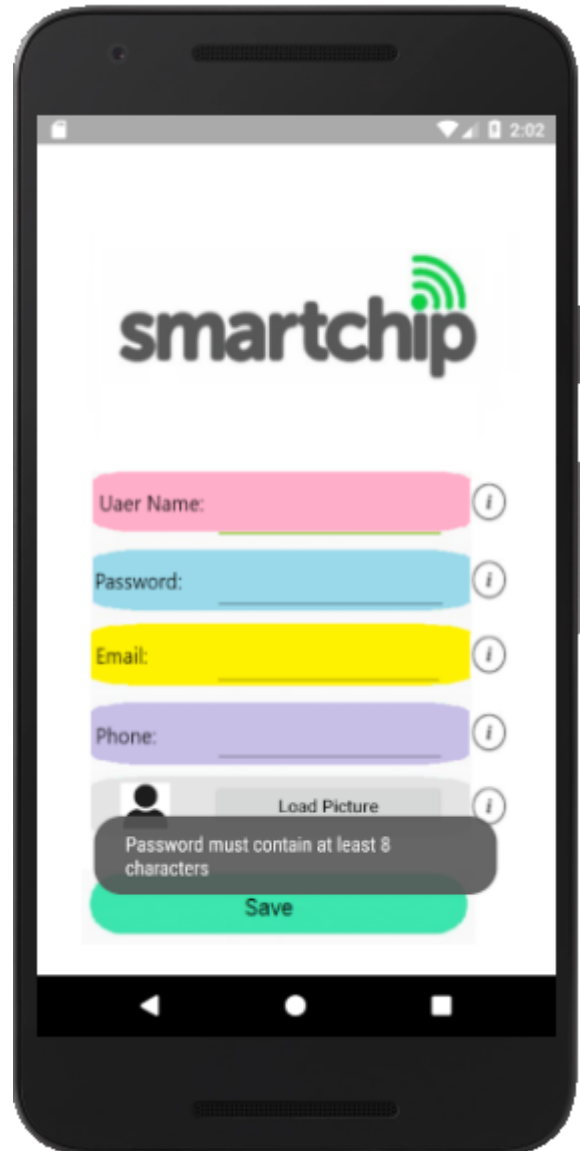
SmartChip – keep it simple

Register screen:



The image shows a smartphone screen displaying the SmartChip registration form. The status bar at the top shows the time as 2:01. The form includes the SmartChip logo at the top, followed by five input fields: 'Uaer Name:' (pink), 'Password:' (light blue), 'Email:' (yellow), 'Phone:' (purple), and a profile picture section with a 'Load Picture' button (grey). Each input field has an information icon (i) to its right. At the bottom of the form is a large green 'Save' button. The Android navigation bar is visible at the very bottom.

Register screen with password information (btnPassInfo clicked):



The image shows the same SmartChip registration form as the previous one, but with a password validation error. The status bar at the top shows the time as 2:02. A grey error message box is overlaid on the 'Password:' field, stating 'Password must contain at least 8 characters'. The 'Save' button remains green and visible at the bottom. The Android navigation bar is visible at the very bottom.

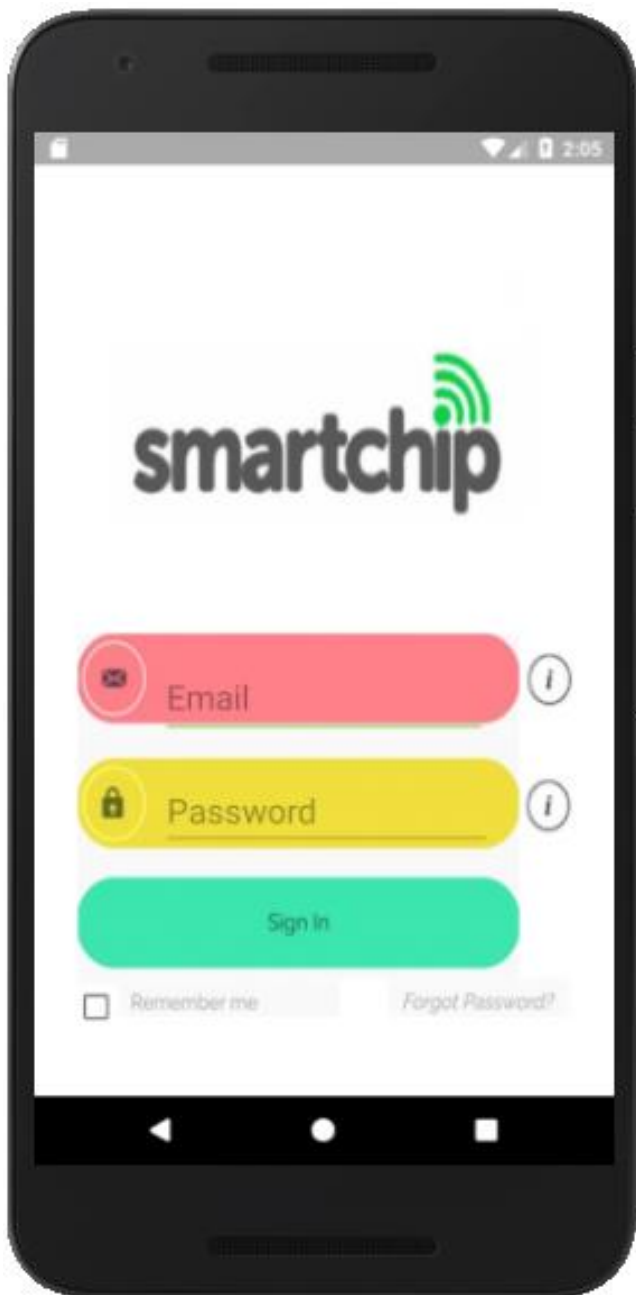
SmartChip – keep it simple

Login Activity:

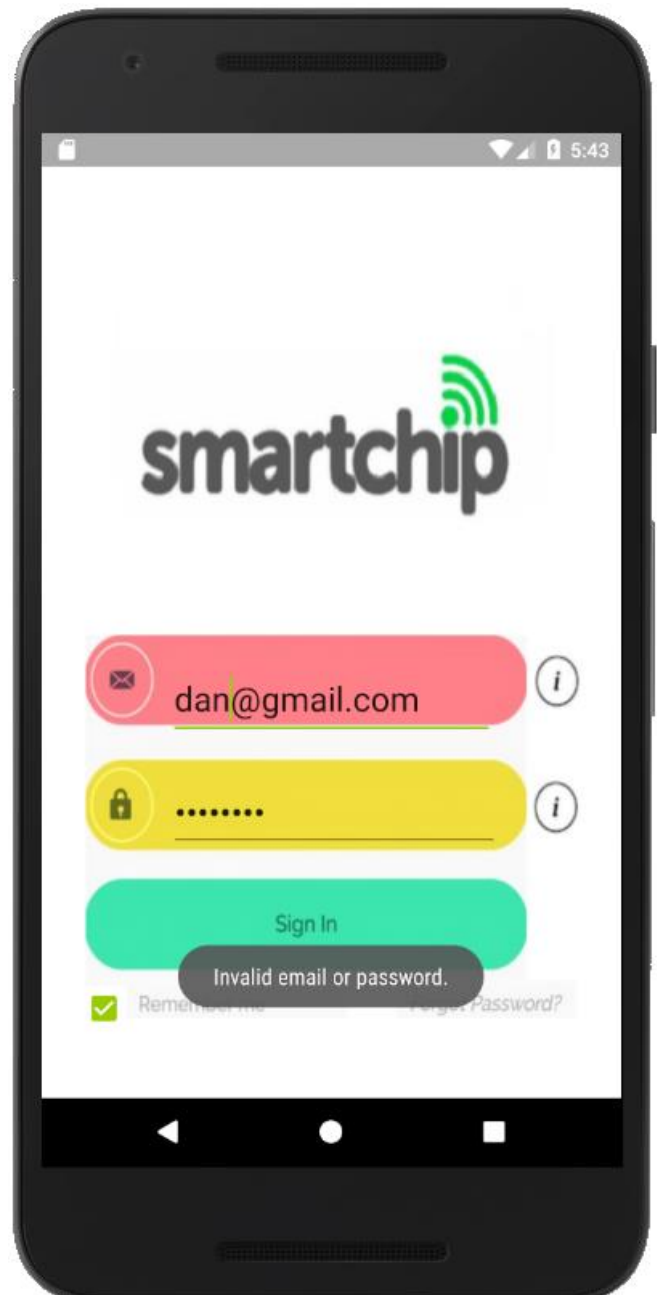
- Textbox to enter the **Email Address** – includes validation of the address, a proper message will be displayed. (not valid mail)
- Textbox to enter the **Password** – includes validation of the password, a proper message will be displayed. (shorter than 8 chars)
- Buttons for **Info** about each data that the user needs to insert:
 - **Email address info:** " You must enter a valid email with @ that registered to the app".
 - **Password info:** "Password must contain at least 8 characters".
- Checkbox which indicates weather the user wants to **be remembered** (next time he opens the app it will save the email and password of the user automatically) if checked the user will be remembered, else the user will have to Sign-in again.
- Link to click on in case the user **forgot his Password** – will transfer the user to the '**ForgettPasswordActivity**'.
- Button to **Sign in** with the credentials the user entered above. (a proper error message will be displayed in case of wrong credentials).

SmartChip – keep it simple

Login screen:



Login screen with wrong input:



SmartChip – keep it simple

Forget Password Activity:

- Textbox to enter the **Email Address** of the user to Reset his Password, if the Email entered is not registered - a proper message will be displayed.

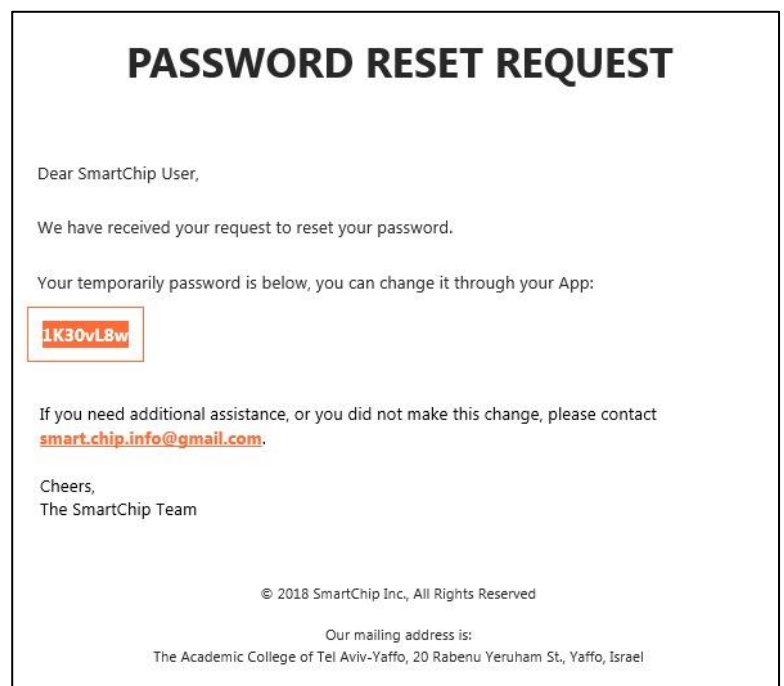
If the Email entered is registered, an email will be sent to the user with a temporary password that the user can change in the future.

- Button to **Send** the user's reset password request.

Forget password screen:



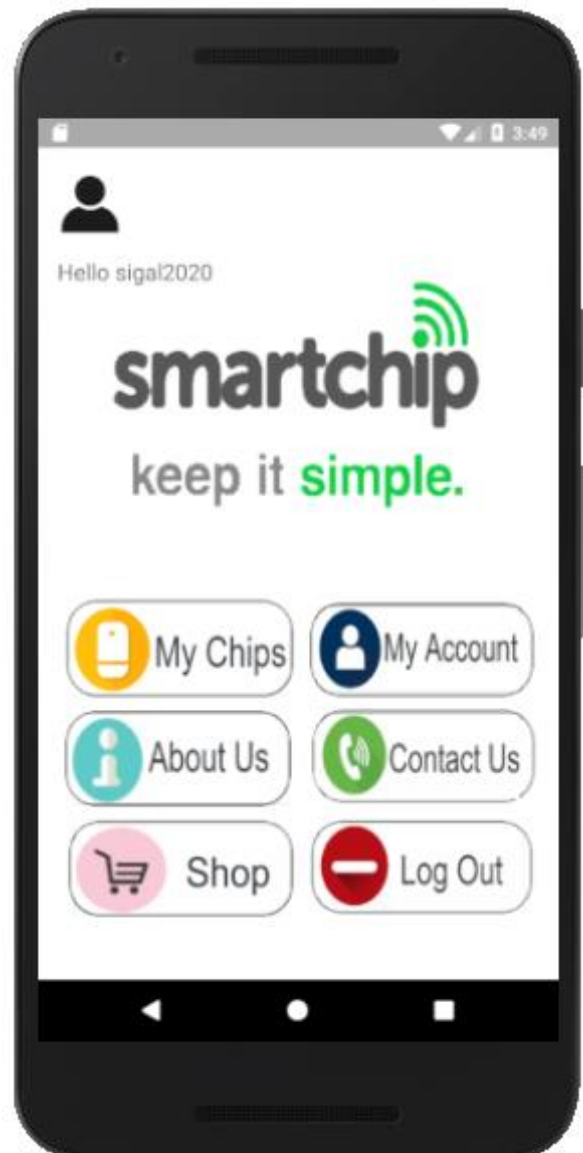
The email the user receives with a temporary password to sign in to the app:



SmartChip – keep it simple

Main Activity and Navigation menu:

- **Orange** Button "**My Chips**" – a click on it will transfer the user to the '**ManageUserChips**' Activity.
- **Blue** Button "**My Account**" – a click on it will transfer the user to the '**ManageUserAccountActivity**'.
- **Light Blue** Button "**About Us**" – a click on it will transfer the user to the '**AboutActivity**'.
- **Green** Button "**Contact Us**" – a click on it will transfer the user to the '**ContactUsActivity**'.
- **Pink** Button "**Shop**" – a click on it will transfer the user to the '**ShopActivity**'.
- **Red** Button "**Log Out**" – a click on it will transfer the user to the '**LoginActivity**'.
- In the left of the top screen – Image view with the <User Profile Image> which displays in all the screens since the user signs in.
- In the left of the top screen – Text view with a message of "**Hello <user name>**" which displays in all the screens since the user signs in.



SmartChip – keep it simple

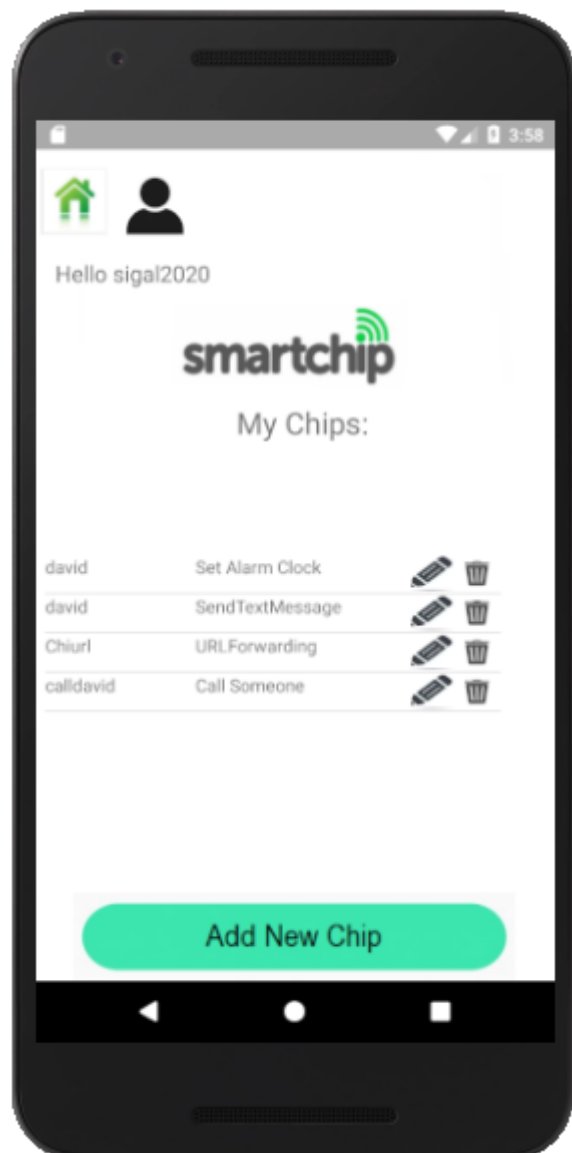
Manage User Chips Activity:

- In the left of the top screen – "**Home**" button – a click on it will transfer the user to '**MainActivity**'.
- In the left of the top screen – Image view with the <User Profile Image> which displays in all the screens since the user sings in.
- In the left of the top screen – Text view with a message of "**Hello <user name>**" which displays in all the screens since the user sings in.
- List View with all the **User chips** details (chip name, chip action).
For each chip there is a possibility to edit the chip information and delete (button **delete**).

Click on button **edit** will transfer the user to the '**EditChipActivity**'.

Click on button **delete** will delete the chip and transfer the user to the '**ManageUserChipActivity**'.

- Button "**Add New Chip**" - a click on it and a dialog with an select options will pop up: only for app user / for everyone. And then it will transfer the user to the '**AddNewUserChipActivity**'.



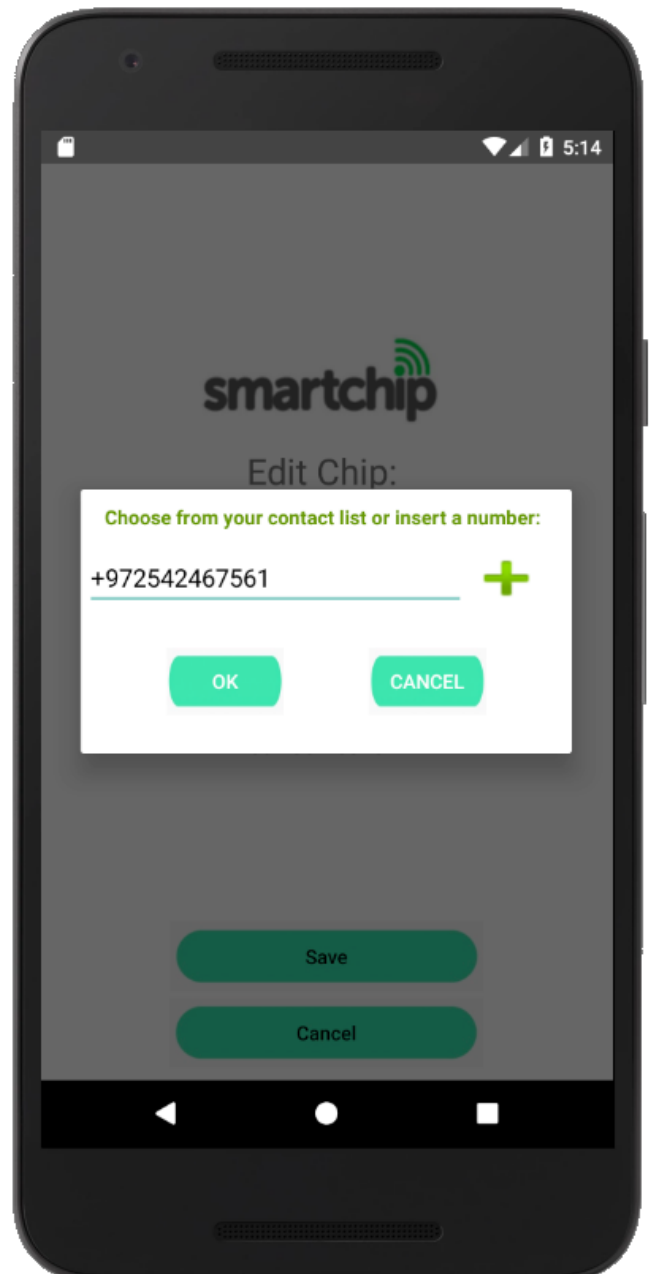
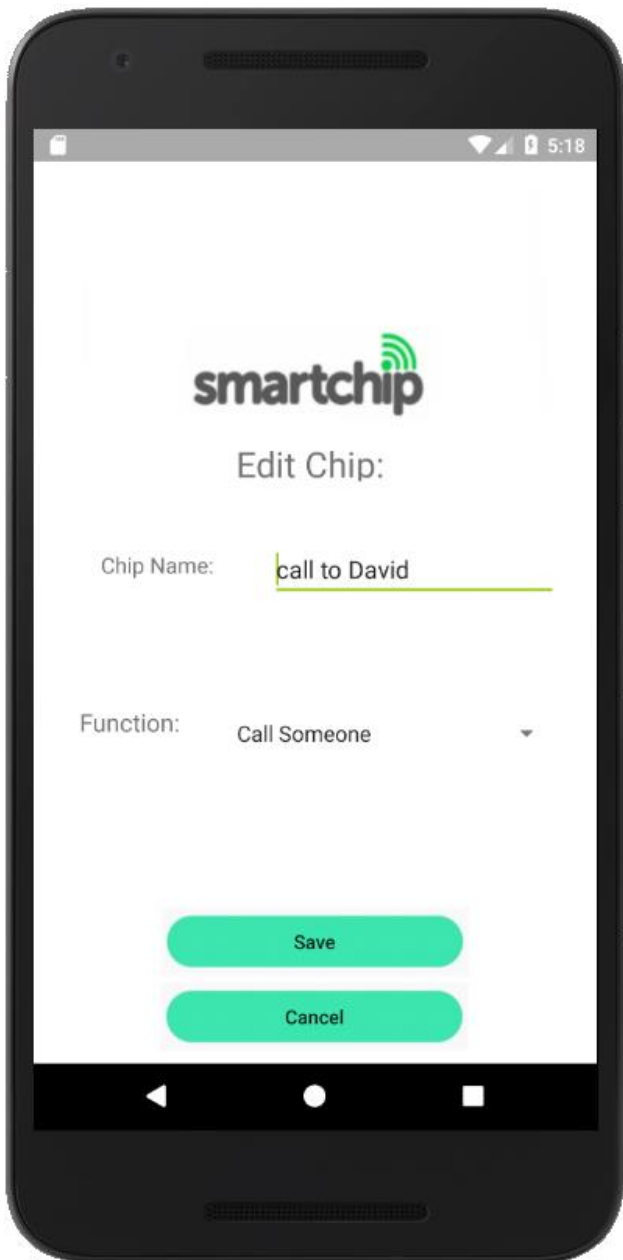
SmartChip – keep it simple

Edit Chip Activity:

- Edit text to set **chip name**.
 - Spinner with options for the **chip action**.
Following the user's selection, the user will be transfer to the appropriate data entry page (phone number, enter a default SMS...).
- Action spinner:
- Call to phone.
 - Send text message.
 - forward to a URL page.
 - Set alarm clock.
 - Set timer.
 -
- Button to **Save** – a click on it will update the user's chip info, in case of an error a proper message will be displayed.
 - Button **Cancel** – a click on it will transfer the user back to '**ManageUserChipsActivity**' and don't change the user chip details.

SmartChip – keep it simple

Edit chip screens (chip with action of 'call someone'):



SmartChip – keep it simple

Add New User Chip Activity:

- In the left of the top screen – "**Home**" button – a click on it will transfer the user to '**MainActivity**'.
- In the left of the top screen – Image view with the <User Profile Image> which displays in all the screens since the user sings in.
- In the left of the top screen – Text view with a message of "**Hello <user name>**" which displays in all the screens since the user sings in.
- Edit text to set **chip name**.
- Spinner with options for the **chip action**.

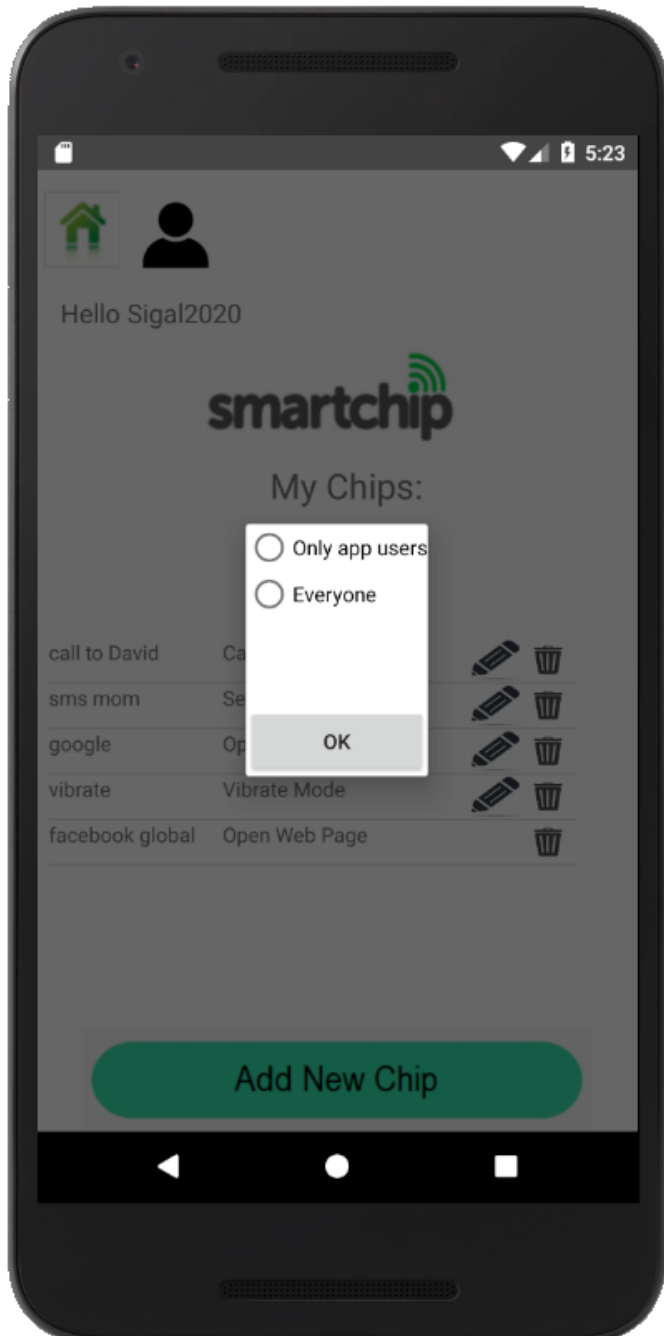
Following the user's selection, the user will be transfer to the appropriate data entry page (phone number, enter a default SMS...).

Action spinner:

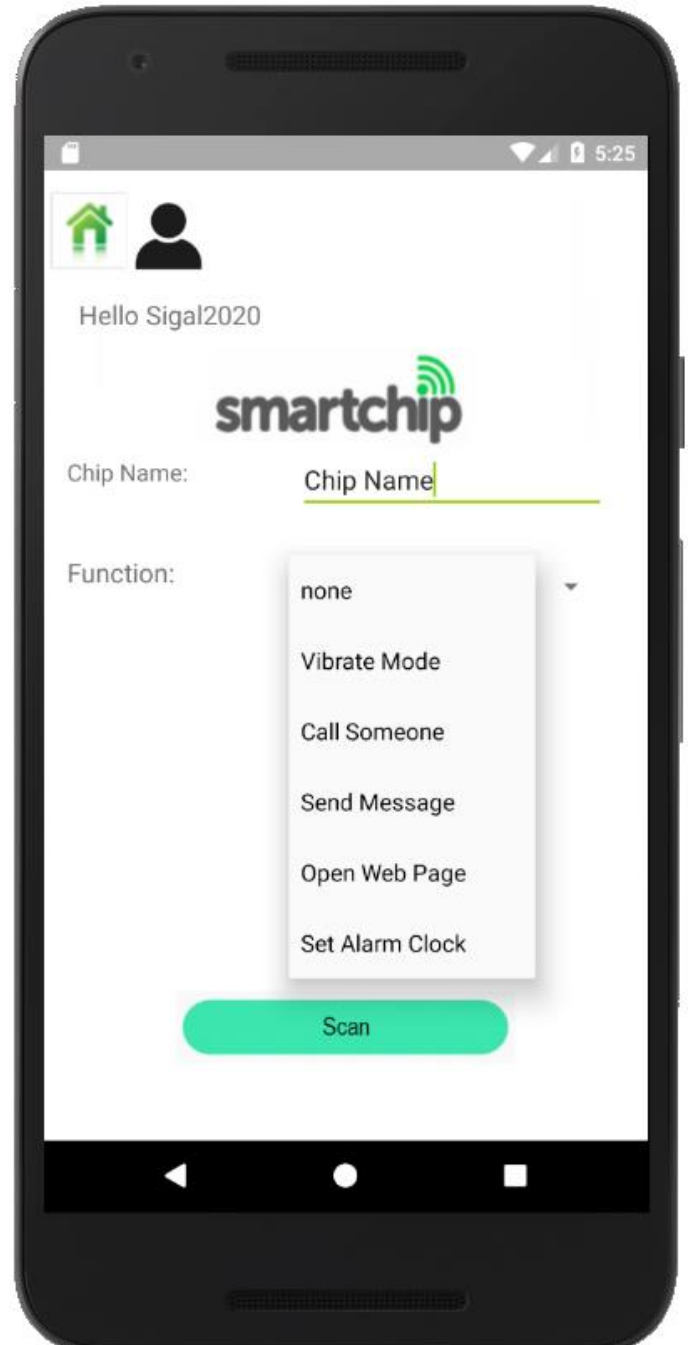
- Call to phone.
 - Send text message.
 - forward to a URL page.
 - Set alarm clock.
 - Set timer.
 -
- Button to **Scan** the new chip – a click on it will transfer the user to '**WriteNfcTagActivity**'.

SmartChip – keep it simple

Add new user chip first screen
(app / global chip):



Add new user chip second screen
(spinner for select action was
clicked):



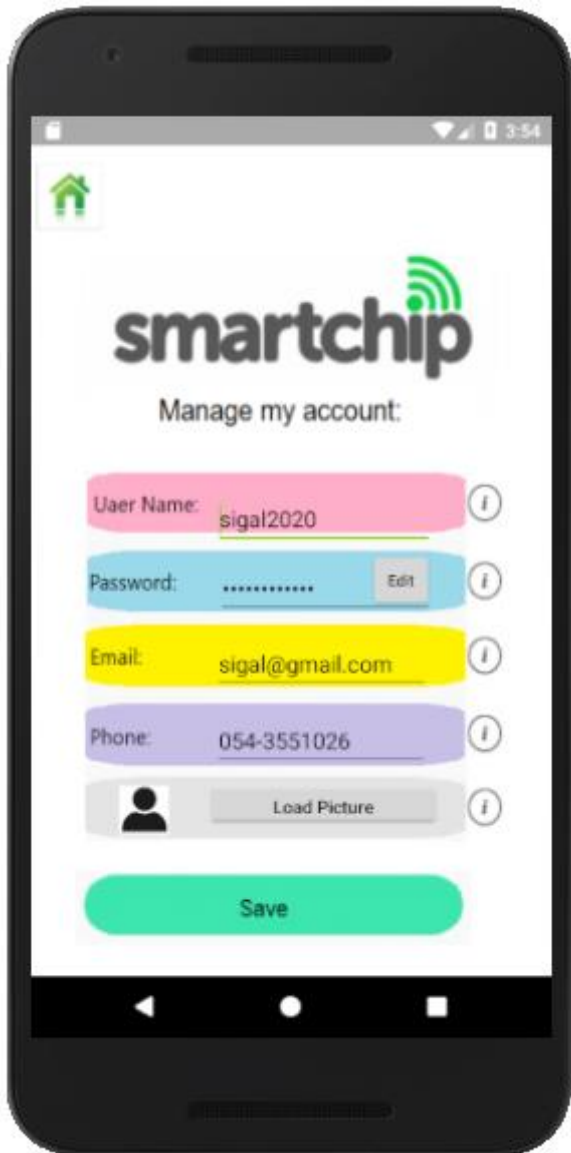
SmartChip – keep it simple

Manage User Account Activity:

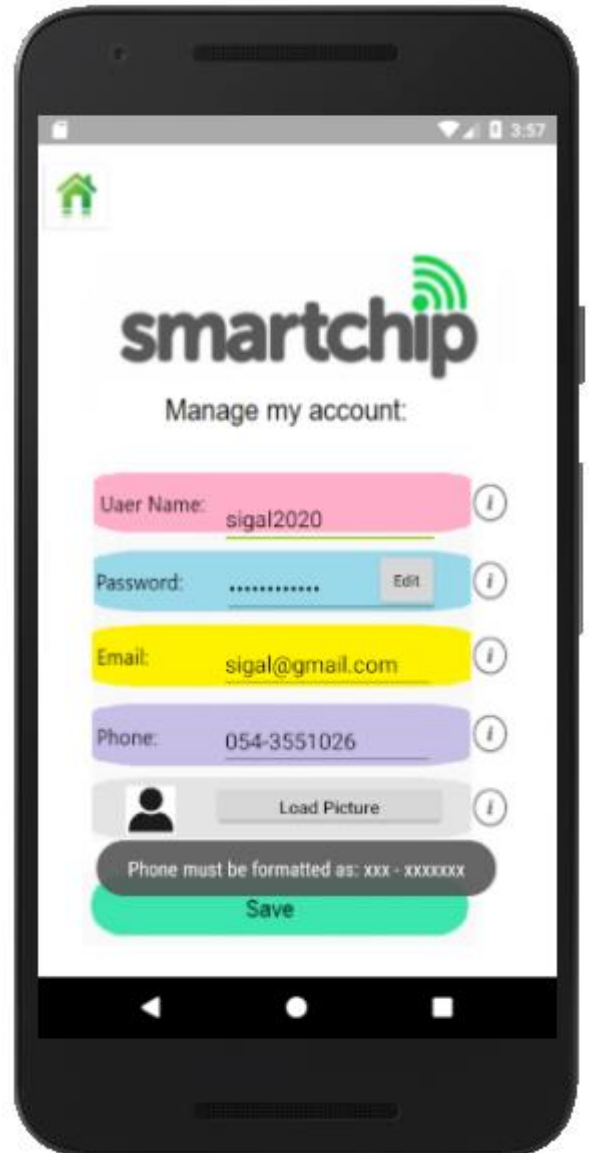
- In the left of the top screen – "**Home**" button – a click on it will transfer the user to '**MainActivity**'.
- In the left of the top screen – Image view with the <User Profile Image> which displays in all the screens since the user signs in.
- In the left of the top screen – Text view with a message of "**Hello <user name>**" which displays in all the screens since the user signs in.
- Label with the **user name**.
- Label with the **Password** (display at ****) and an **Edit** button – click on it will transfer the user to '**ResetPasswordActivity**'.
- Label with the **Email-Address**.
- Label with the **Mobile Phone** – the number can be updated by the user, it includes validation of the phone, a proper message will be displayed. (10 digits)
- Button to **Select Picture** for the user's profile picture – includes validation of the picture, a proper message will be displayed. (JPEG or PNG and less than 500KB)
If no picture were selected, the user's profile picture will be displayed.
- Button "**Save**" – a click on it will update the user's personal info, in case of an error a proper message will be displayed.

SmartChip – keep it simple

My account screen:



My account screen with phone number info (btnPhoneInfo clicked):



SmartChip – keep it simple

Reset Password Activity:

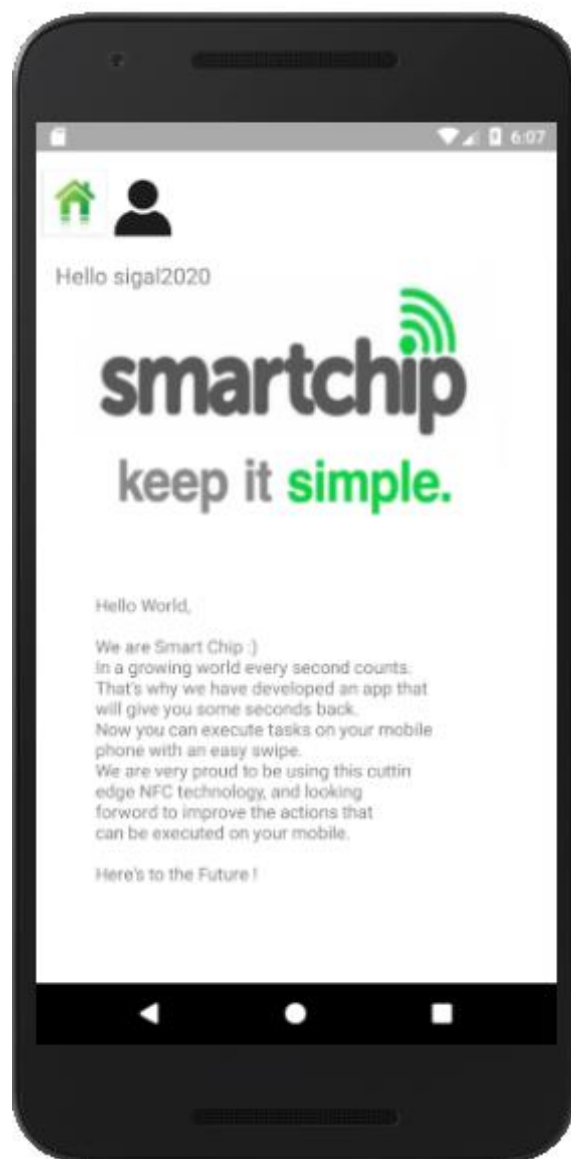
- In the left of the top screen – Image view with the **<User Profile Image>** which displays in all the screens since the user signs in.
- In the left of the top screen – Text view with a message of "**Hello <user name>**" which displays in all the screens since the user signs in.
- Edit text to enter the **current password**.
- Edit text to enter the **new password**.
- Button **Save** – a click on it will save the user new password and return the user to '**ManageUserAccountActivity**'.
- Button **Cancel** – a click on it will transfer the user to '**ManageUserAccountActivity**', and don't change the password.



SmartChip – keep it simple

About us Activity:

- In the left of the top screen – "**Home**" button – a click on it will transfer the user to '**MainActivity**'.
- In the left of the top screen – Image view with the <User Profile Image> which displays in all the screens since the user sings in.
- In the left of the top screen – Text view with a message of "**Hello <user name>**" which displays in all the screens since the user sings in.
- Text view with "**about us**" section.



SmartChip – keep it simple

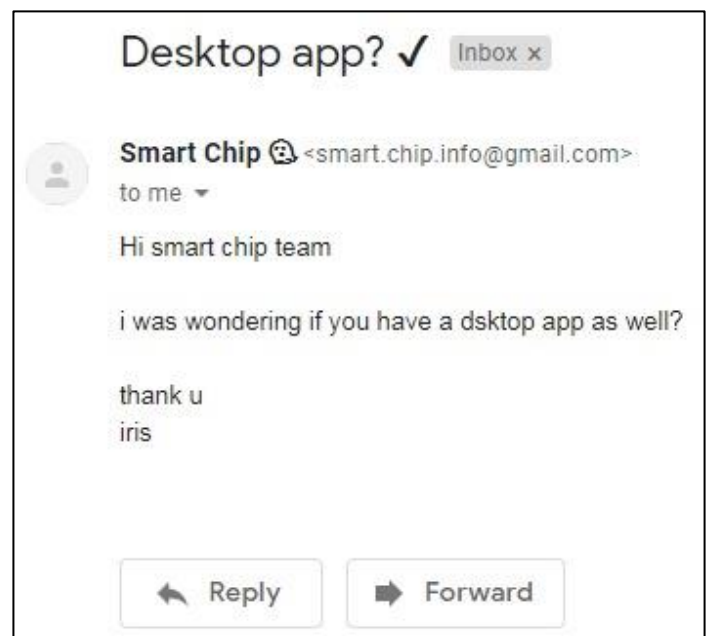
Connect us Activity:

- In the left of the top screen – "**Home**" button – a click on it will transfer the user to '**MainActivity**'.
- In the left of the top screen – Image view with the <User Profile Image> which displays in all the screens since the user sings in.
- In the left of the top screen – Text view with a message of "**Hello <user name>**" which displays in all the screens since the user sings in.
- Edit text for insert the **subject**.
- Edit text for insert the **message** to send.
- Button **Send** – a click on it will send the mail to smart.chip.info@gmail.com and will transfer the user to '**MainActivity**'.

Contact us screen:



The email the user sent through the app:

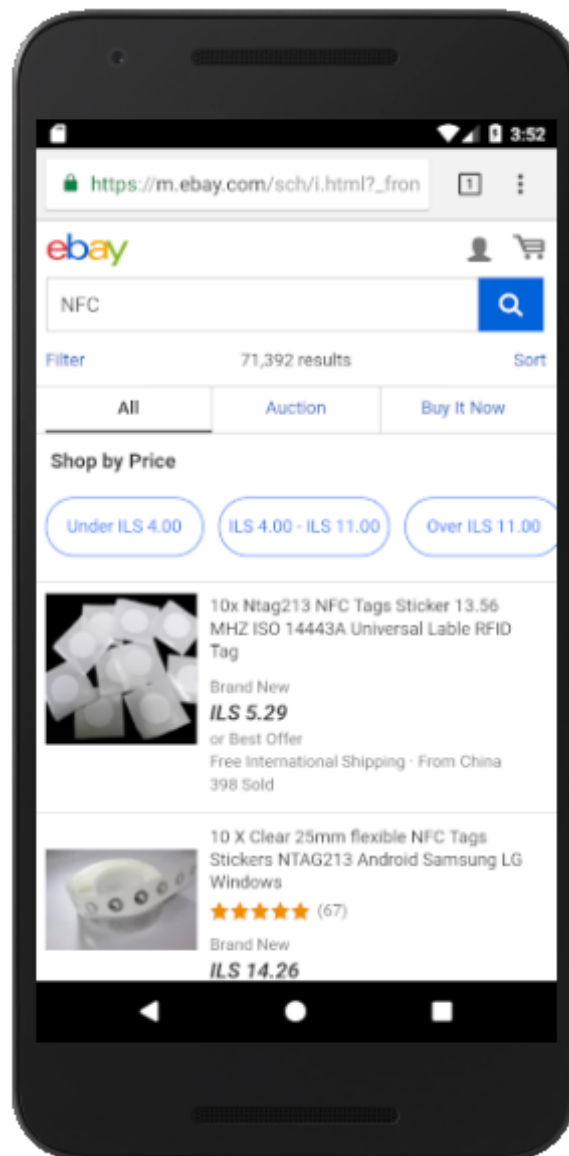


SmartChip – keep it simple

Shop Activity:

- Web view that transfer the user to a Ebay page:

["https://www.ebay.com/sch/i.html?_from=R40&_trksid=m570.l1313&_nkw=NFC&_sacat=0"](https://www.ebay.com/sch/i.html?_from=R40&_trksid=m570.l1313&_nkw=NFC&_sacat=0)



Desktop app:

Application functionality:

Login Form:

- Textbox to enter the **Email Address** – includes validation of the address, a proper message will be displayed. (not valid mail)
- Textbox to enter the **Password** – includes validation of the password, a proper message will be displayed. (shorter than 8 chars).
- Link label to click on in case the user forgot his Password – will transfer the user to the 'ForgettPasswordForm'.
- Button to **Register** for new users. When hovering with the mouse on the button it is painted **green**. Clicked on it will transfer the user to the 'RegisterForm'.
- Button to **Login** with the details the user entered above. (a proper error message will be displayed in case of wrong credentials). When hovering with the mouse on the button it is painted **green**. Clicked on it (with valid credentials) will transfer the user to the 'UserMainForm'.
- Button to **Exit**. When hovering with the mouse on the button it is painted **red**. Clicked on it will close the app.

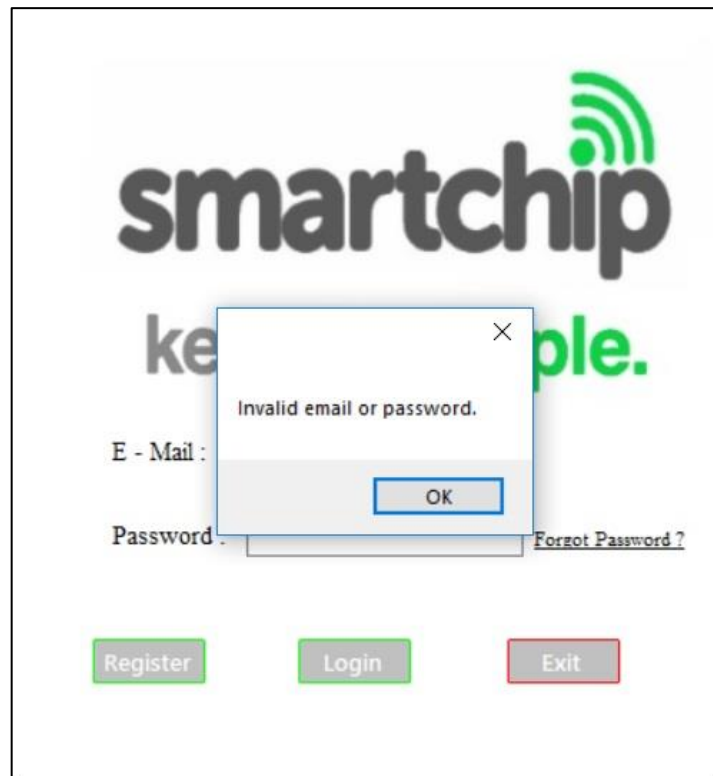
SmartChip – keep it simple

Login Form:



The login form features the SmartChip logo at the top, which includes a green Wi-Fi symbol above the word "smartchip" in a bold, dark grey font, followed by the tagline "keep it simple." in a smaller font, with "simple." in green. Below the logo, there are two input fields: "E - Mail :" and "Password :". To the right of the password field is a link that says "Forgot Password ?". At the bottom of the form, there are three buttons: "Register" (green), "Login" (green), and "Exit" (red).

Login Form with invalid details:



This image shows the same login form as above, but with an error message displayed in a small dialog box. The dialog box has a title bar with a close button (X) and contains the text "Invalid email or password." Below the text is an "OK" button. The background of the login form is slightly faded, and the "E - Mail :" and "Password :" labels are visible behind the dialog box.

SmartChip – keep it simple

Register Form:

- Textbox to enter the **User Name** – includes validation of the length, a proper message will be displayed. (at less 3 character).
- Textbox to enter the **Phone Number** - includes validation of the phone, a proper message will be displayed. (10 digits)
- Textbox to enter the **Email Address** – includes validation of the address, a proper message will be displayed. (not valid mail)
- Textbox to enter the **Password** – includes validation of the password, a proper message will be displayed. (shorter than 8 chars).
- Button to **Upload Image**. When hovering with the mouse on the button it is painted green. Clicked on it will transfer the user to the browse photo of his computer.
- Picture box for **User Photo**.
- Button to **Register** for new users. When hovering with the mouse on the button it is painted green. Clicked on it will transfer the user to the 'RegisterForm'.
- Button **Done**. When hovering with the mouse on the button it is painted green. Clicked on it will transfer the user to the 'UserMainForm'.
- Button **Back**. When hovering with the mouse on the button it is painted black. Clicked on it will transfer the user to the 'LoginForm'.
- Image button **Exit**. Clicked on it will close the form.



The screenshot shows a web form titled 'smartchip keep it simple.' with a logo featuring a green Wi-Fi symbol. The form contains the following fields and controls:

- User Name :** Textbox with the value 'Sigal'.
- Phone Number :** Textbox with the value '054-5555555'.
- Email :** Textbox with the value 'sigal@gmail.com'.
- Password :** Password textbox with masked characters '*****'.
- Photo :** A section containing an 'Upload Photo' button (highlighted in green) and a circular placeholder image of a woman.
- Navigation:** A 'Back' button (grey) and a 'Done' button (green) at the bottom.

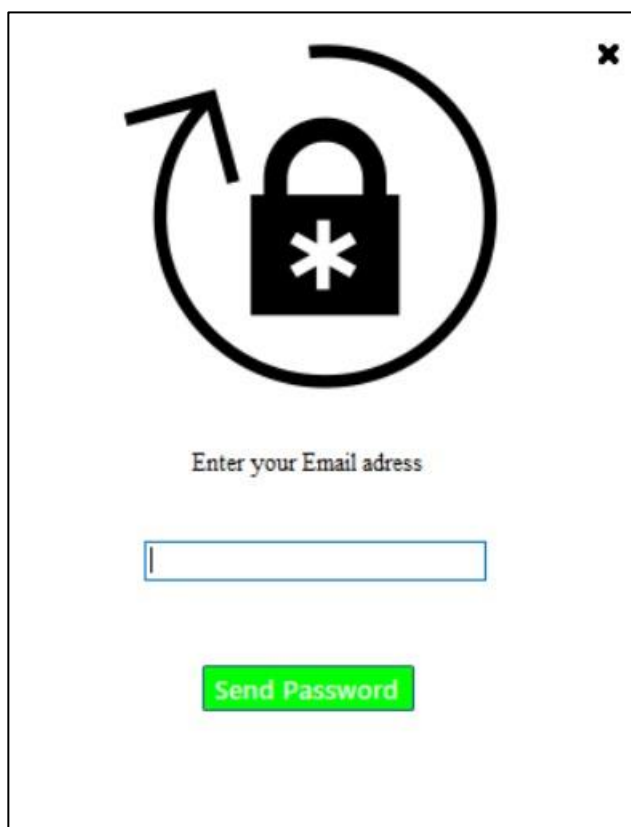
The form is enclosed in a window with a green header bar and a close button (X) in the top right corner.

SmartChip – keep it simple

Forget Password Form:

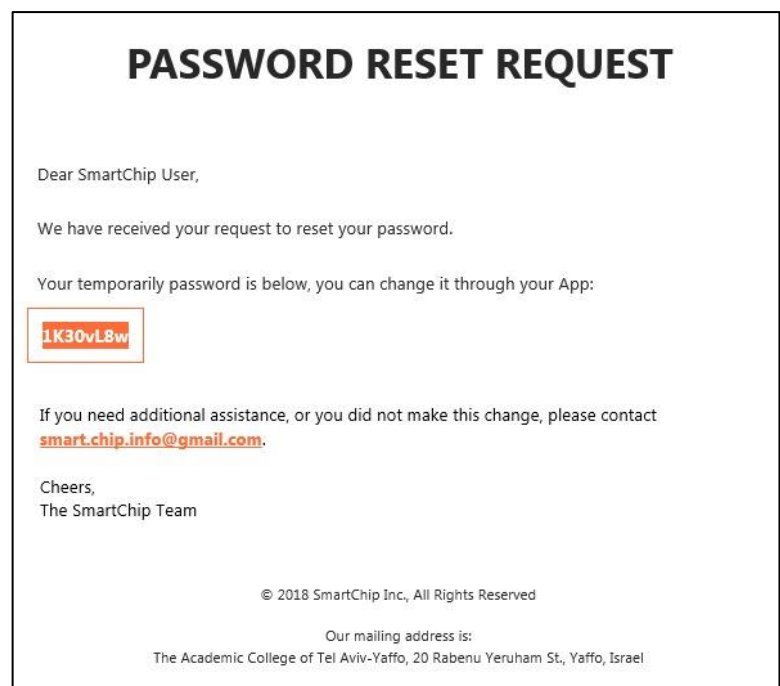
- Textbox to enter the **Email Address** of the user to Reset his Password, if the Email entered is not registered - a proper message will be displayed.
If the Email entered is registered, an email will be sent to the user with a temporary password that the user can change in the future.
- Button to **Send** the user's reset password request. When hovering with the mouse on the button it is painted **green**.
- Image button **Exit**. Clicked on it will close the form.

Forget password Form:



The image shows a 'Forget password' form. At the top, there is a large circular icon containing a padlock with an asterisk inside, and a curved arrow pointing upwards and to the left. In the top right corner of the form is a small 'x' icon. Below the icon, the text 'Enter your Email address' is displayed. Underneath this text is a rectangular input field. At the bottom of the form is a green button with the text 'Send Password' in white.

The email the user receives with a temporary password to sign in to the app:



The image shows the content of an email titled 'PASSWORD RESET REQUEST'. The text inside the email is as follows:

Dear SmartChip User,

We have received your request to reset your password.

Your temporarily password is below, you can change it through your App:

1K30vL8w

If you need additional assistance, or you did not make this change, please contact smart.chip.info@gmail.com.

Cheers,
The SmartChip Team

© 2018 SmartChip Inc., All Rights Reserved

Our mailing address is:
The Academic College of Tel Aviv-Yaffo, 20 Rabenu Yeruham St., Yaffo, Israel

SmartChip – keep it simple

User Main Form:

- Image button **Exit**. Clicked on it will close the form.
- Panel that contains 4 different forms:
 - Chip Table From.
 - About Us Form.
 - Shop Form.
 - Contact Us Form.

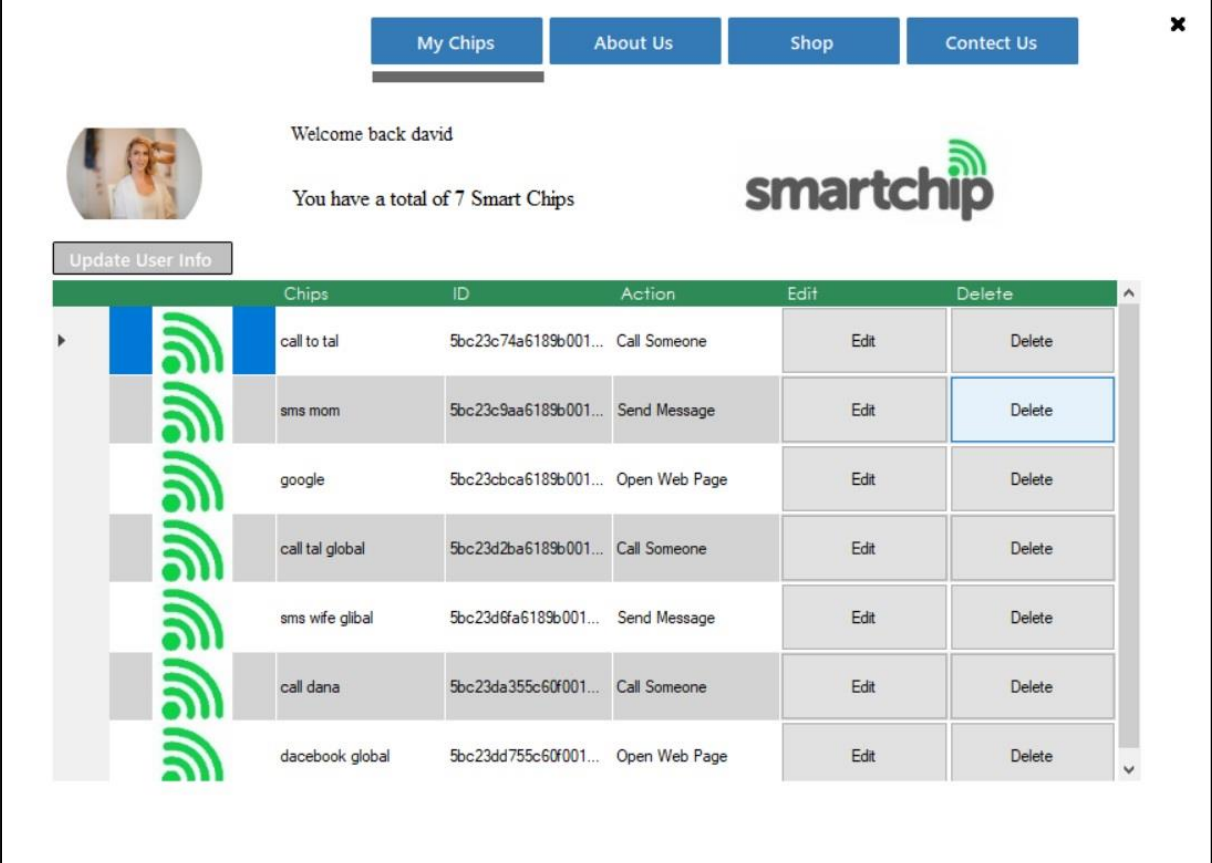
with every click on one of the blue Tab buttons the appropriate form will be loaded to the panel.

- Button **My Chips**. Clicked on it will painted the under tab to gray and load the 'ChipTableFrom' to the panel.
- Button **About Us**. Clicked on it will painted the under tab to gray and load the 'AboutUsForm' to the panel.
- Button **Shop**. Clicked on it will painted the under tab to gray and load the 'ShopFrom' to the panel.
- Button **Contact Us**. Clicked on it will painted the under tab to gray and load the 'ContactUsFrom' to the panel.

SmartChip – keep it simple

- Chip Table Form:

- Image button **Exit**. Clicked on it will close the form.
- Picture box for **User Photo**.
- Button **Update User Info**. Clicked on it will painted it to black and open the 'UpdateUserForm'.
- **Welcome back <user name>** Label.
- **< Chips number>** Label.
- Data Grid with all the user chips data. For each row there are 2 buttons:
 - **Edit** button. Click on it open the 'EditChipForm'.
 - **Delete** Button. Clicked on it present a message that the chip has been deleted and update the Data Grid of the chips.



The screenshot displays the SmartChip application interface. At the top, there is a navigation bar with buttons for 'My Chips', 'About Us', 'Shop', and 'Contact Us'. Below the navigation bar, the user profile section shows a welcome message 'Welcome back david' and a notification 'You have a total of 7 Smart Chips'. A 'smartchip' logo is also present. A button labeled 'Update User Info' is visible. The main content area features a table with columns: Chips, ID, Action, Edit, and Delete. The table lists seven chips, each with a unique ID and an associated action. The 'Delete' button for the second chip, 'sms mom', is highlighted in blue.

Chips	ID	Action	Edit	Delete
call to tal	5bc23c74a6189b001...	Call Someone	Edit	Delete
sms mom	5bc23c9aa6189b001...	Send Message	Edit	Delete
google	5bc23cbca6189b001...	Open Web Page	Edit	Delete
call tal global	5bc23d2ba6189b001...	Call Someone	Edit	Delete
sms wife glibal	5bc23d6fa6189b001...	Send Message	Edit	Delete
call dana	5bc23da355c60f001...	Call Someone	Edit	Delete
dacebook global	5bc23dd755c60f001...	Open Web Page	Edit	Delete

SmartChip – keep it simple

- Edit Chip Form:

- Image button **Exit**. Clicked on it will close the form.
- Label for the selected **chip details**.
- Text box for choose a **Chip Name**.
- Label that appears only **if the chip is global**, with "This is a global chip, could not edit chip" and colored in **red**.
- Combo box for choose the **Chip Action**:
 - Send message. A click on it will present Textbox for insert the **Phone number** and a Rich Text for insert the **Message Body**.
 - Open web page. A click on it will present Textbox for insert the **URL**.
 - Call someone. A click on it will present a Textbox for insert **Phone number**.
 - Set alarm clock. A click on it will present **Time picker**.
 - Vibrate mode.
- Button **Done**. When hovering with the mouse on the button it is painted **green**. Clicked on it will change the contain of the chip, close this form and update chips details in the Data Grid.
- Button **Exit**. When hovering with the mouse on the button it is painted **red**. Clicked on it will ask the user if he sure and if yes, it will close this form without save the changes.

Edit Chip Form
that 'Send
Message' action
was selected in
the Combo box:

The screenshot shows a window titled "Edit Chip" with a close button (X) in the top right corner. The form displays the following information:

- Chip Name: google
- Id: 5bc23cbca6189b001577bb7a
- Action: Open Web Page

Below this information are two sections:

- ChangeAction:** A dropdown menu is open, showing the following options: "Send Message" (highlighted), "Send Messege", "Open Web Page", "Call Someone", "Set Alarm Clock", and "Set Timer".
- Change name:** A text box containing the word "google".

Below these sections are two more fields:


- Send To :** A text box.
- Message Body :** A large text area.

At the bottom of the form are two buttons:

- Done:** A green button.
- Exit:** A red button.

SmartChip – keep it simple

- Update User Form:
 - Image button **Exit**. Clicked on it will close the form.
 - Text box for **User Name**.
 - Text box for **User Email**.
 - Text box for **User Phone Number**.
 - Button **Upload Photo**. When hovering with the mouse on the button it is painted green. Clicked on it will open the browser to upload a photo.
 - Picture box for **User Photo**.
 - Button **Change Password**. When hovering with the mouse on the button it is painted black. Clicked on it will open the 'ChangePasswordForm'.
 - Button **Done**. When hovering with the mouse on the button it is painted green. Clicked on it will update the user details and close this form.
 - Button **Exit**. When hovering with the mouse on the button it is painted red. Clicked on it will ask the user if he sure and if yes, it will close this form without save the changes.



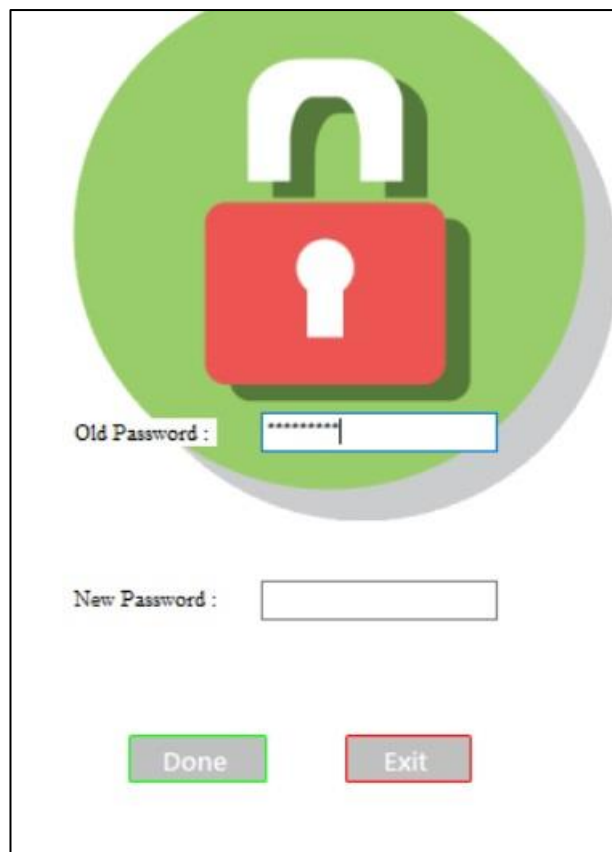
The screenshot shows a software window titled "SmartChip" with a green header bar and a close button (X) in the top right corner. The form contains the following elements:

- Name :** A text box containing the text "david".
- Email :** A text box containing the text "david@gmail.com".
- Phone :** A text box containing the text "054-2467561".
- Buttons:**
 - Upload Photo:** A green button.
 - Change Password:** A grey button.
 - Done:** A green button.
 - Exit:** A red button.
- Icons:**
 - A blue circular icon with a white person silhouette.
 - A large grey 3D person icon.
 - A blue circular icon with a white lowercase "i" (information).

SmartChip – keep it simple

- Change Password Form:

- Textbox to enter the **Old Password**.
- Textbox to enter the **new password**.
- Button **Done**. When hovering with the mouse on the button it is painted **green**. A click on it will save the user new password and close this form.
- Button **Exit**. When hovering with the mouse on the button it is painted **red**. A click on it will close this form without saving the changes.

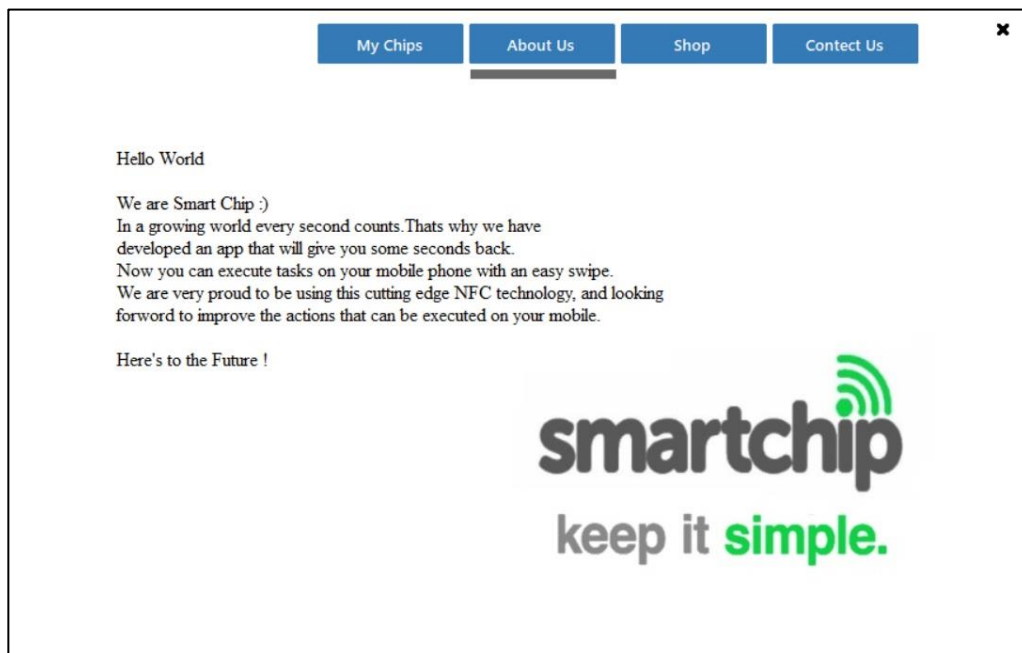


The image shows a user interface for a 'Change Password' form. At the top center is a large green circle containing a white padlock icon with a red keyhole. Below this graphic, the text 'Old Password :' is followed by a text input field containing several asterisks. Further down, the text 'New Password :' is followed by an empty text input field. At the bottom of the form are two buttons: 'Done' with a green border and 'Exit' with a red border.

SmartChip – keep it simple

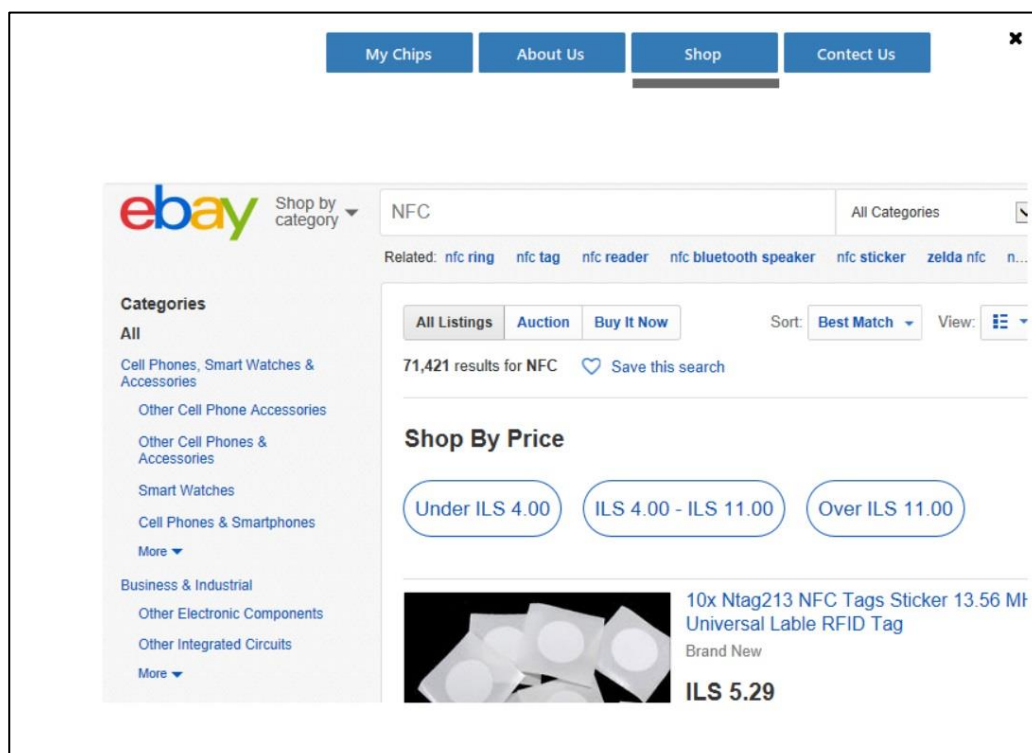
- About Us Form:

- Image button **Exit**. Clicked on it will close the form.
- Label with "**about us**" section.



- Shop Form:

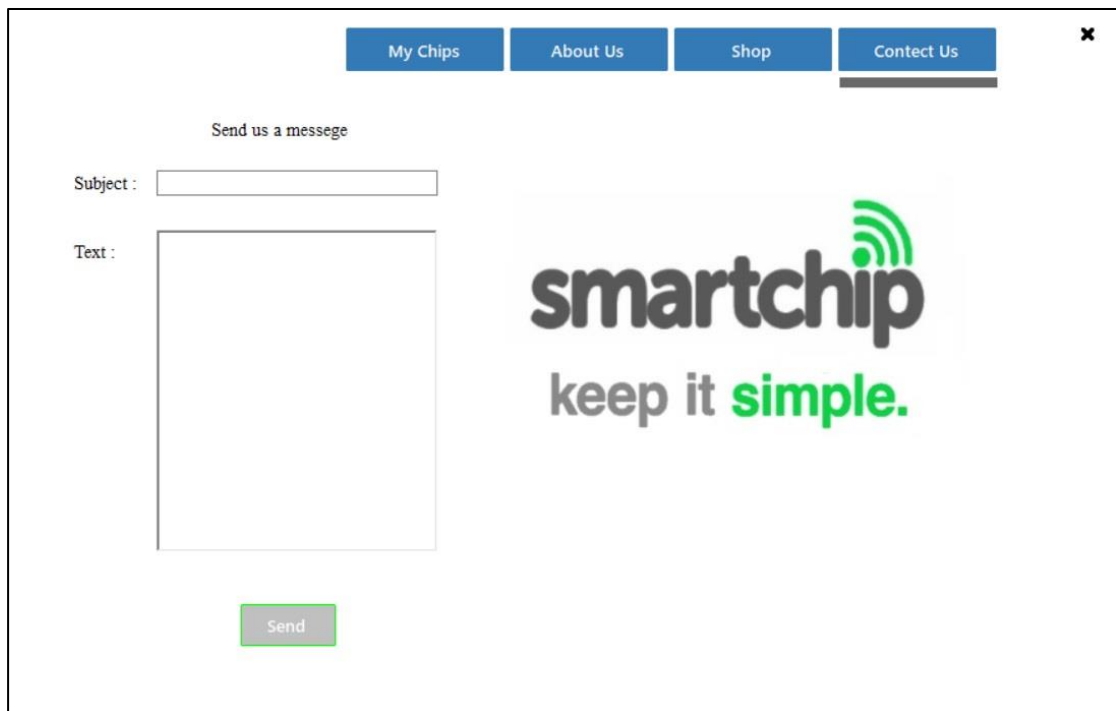
- Web browser that display this Ebay page:
["https://www.ebay.com/sch/i.html? from=R40& trksid=m570.l1313& nkw=NFC& sacat=0"](https://www.ebay.com/sch/i.html?from=R40&trksid=m570.l1313&nkw=NFC&sacat=0)



SmartChip – keep it simple

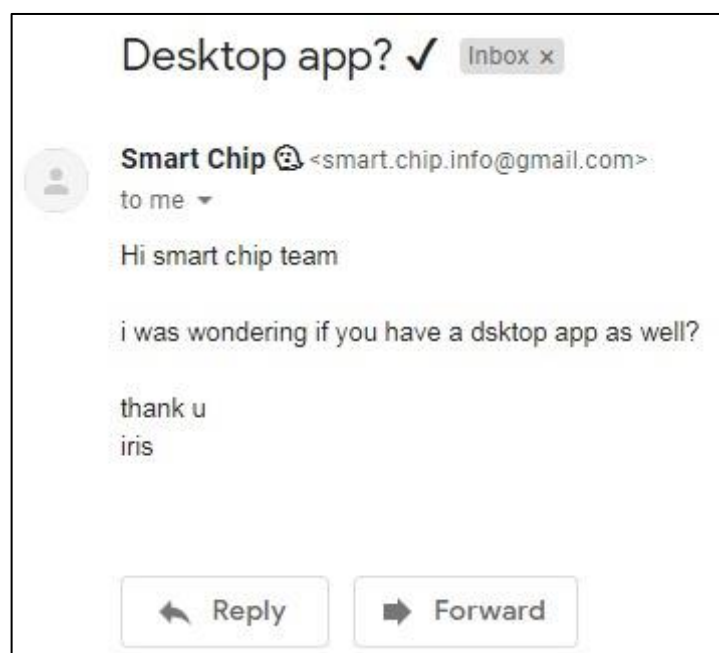
- Contact Us Form:

- Image button **Exit**. Clicked on it will close the form.
- Text box for the **Subject**.
- Rich Textbox for the **Message Body**.
- Button **Send**. When hovering with the mouse on the button it is painted **green**. A click on it will send the mail to smart.chip.info@gmail.com and will close this form.



The screenshot shows a web interface for the 'SmartChip' application. At the top, there is a navigation bar with four buttons: 'My Chips', 'About Us', 'Shop', and 'Contact Us'. The 'Contact Us' button is highlighted with a dark blue underline. Below the navigation bar, the text 'Send us a message' is centered. On the left side, there is a form with two fields: 'Subject :' followed by a text input box, and 'Text :' followed by a larger rich text area. Below the text area is a green 'Send' button. On the right side, there is a large logo for 'smartchip' with the tagline 'keep it simple.' in green. The logo features a green Wi-Fi symbol above the word 'smartchip'.

The email the user sent through the app:



SmartChip – keep it simple

Architecture and Implementation overview:

Client ↔ Server architecture. HTTP Requests see [here](#).

Server code - [here](#)

Overview: Fast, asynchronous, scalable and secure RESTful services with Node, Express and MongoDB.

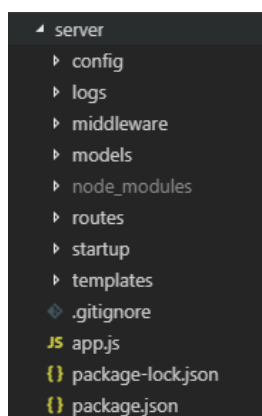
Server technologies and tools:

- NodeJs - An asynchronous event driven JavaScript runtime server-side framework, designed to build scalable network applications.
- MongoDB - NoSQL distributed database that stores data in flexible, JSON-like documents.
- Mongoose - Mongoose is a MongoDB object modeling tool designed to work in an asynchronous environment.
- Heroku - Cloud platform as a service (PaaS), for our Express server deployment.
- AWS - Amazon Web Service, used for our MongoDB server deployment.
- Express - Minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications.
- Jsonwebtoken - This module lets you authenticate HTTP requests using JWT tokens. JWTs are typically used to protect API endpoints.

SmartChip – keep it simple

- Config - Node-config organizes hierarchical configurations for app deployments.
- Winston - Simple and universal logging library with support for multiple transports.
- Joi - Robust validation plugin that validate objects against a predefined object schema.
- fawn - Promise based Library for transactions in MongoDB used for implementation Two-phase commit protocol.
- Bcrypt - based on the Blowfish cipher (symmetric-key algoritam).
- Nodemailer - module for Node.js applications to allow email sending. for reset password and to contact us.
- Helmet - helps to secure Express apps by setting various HTTP headers.
- Compression - compress response bodies for all request that traverse through the middleware.

Directory Structure:



Config – Define custom environment variables for secure reasons that is predefined by us before server deployment.

SmartChip – keep it simple

Logs – This folder contains logs files for the standard server operations, including both unhandled rejection & unhandled exceptions errors using “Winston”.

Middleware – This folder contains our middlewares - functions that have access to the HTTP request object (req), the HTTP response object (res), and the next middleware function in the application’s request-response cycle.

For example:

- **auth.js** middleware for user authentication.
- **admin.js** middleware for user authorization.
- **error.js** middleware for handling versions exceptions.

Models – This folder contain MongoDB models “Schema” definitions. Instance of a model is called a document. Models are responsible for creating and reading documents from the underlying MongoDB database.

Routes – This folder contains our application’s endpoints (URIs) respond to client requests in a Restful API architecture with CRUD operations:

- Create – http POST request
- Read - http GET request
- Update - http PUT request
- Delete - http DELETE

```
GET /api/customers
GET /api/customers/1
PUT /api/customers/1
DELETE /api/customers/1
POST /api/customers
```

Startup – This folder contains initial boot operations before running the application – applying specific routers the Express app, establish connection to MongoDB, initialize “Nodemailer” mail account, define logs files... and so on.

Chain of processes example:

Scenario: Existing user try to add a new ‘Chip’ to his chips list.

SmartChip – keep it simple

1. Http POST request send to our Express server:

REST client interface showing a POST request to `https://smartchip.herokuapp.com/api/chips`.

Headers (2)

Key	Value
<input checked="" type="checkbox"/> x-auth-token	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOiI1YmMy...
<input checked="" type="checkbox"/> Content-Type	application/json

Body

form-data x-www-form-urlencoded raw binary JSON (application/json)

```
1 {
2   "name": "Call To Sigal",
3   "action": "CallSomeone",
4   "options": ["050-1234567"],
5   "isGlobal": "false"
6 }
```

x-auth-token – holding the json web token given by the server when user registered.

Body – contain a json object that will be used to create the new chip.

2. The user http request handling by the authentication & authorization middlewares.

To ensure authentication we decoded the user 'jwt' sending attach to his HTTP POST header using the 'jsonwebtoken' module. Proper message will be sent in case of an Error.

```
JS authjs x
1
2 const jwt = require('jsonwebtoken');
3 const config = require('config');
4
5 module.exports = function (req, res, next) {
6   // Verify user web token
7   const token = req.header('x-auth-token');
8   if (!token) return res.status(401).send('Access denied. No token provided.');
```

```
9   // Try to decoded user token
10  try {
11    const decoded = jwt.verify(token, config.get('jwtPrivateKey'));
12    req.user = decoded; //in router handels we can access --> req.user._id
13    next();
14  } catch (ex) {
15    res.status(400).send('Invalid token.');
```

```
16  }
17 }
```

SmartChip – keep it simple

Then to ensure authorization we check a hidden value 'isAdmin' in the user jwt hash by us when adding the user to the DB.

```
JS admin.js x
1
2 module.exports = function (req, res, next) {
3   //401 Unauthorized - access protected resources with invalid jwt
4   //403 Forbidden - access protected resources without right auth
5   if(!req.user.isAdmin) return res.status(403).send('Access denied.');
```

3. Processing request –

- Use “Joi” module predefine schema to validate user req.
- Use “Mongoose” module to query our MongoDB.
- Create new chip and use other validation define in the “Mongoose” schema.
- Our NoSQL database build in a Dynamic structure – its mean that every user hold an array for object Id reference to a Chip document. So, to handle with an ‘abort’ situation we use ‘Two-Phase-Commit’ protocol impl with “Fawn”

```
chips.js x
38 // Add chip to user
39 router.post('/', [auth, admin], async (req, res) => {
40   // Ensure data validation using 'joi'
41   const { error } = validate(req.body);
42   if (error) return res.status(400).send(error.details[0].message);
43   // Ensure user exist
44   let user = await User.findById(req.user._id); //from json web token
45   if (!user) return res.status(404).send('The user with the given TOKEN not found.');
```

SmartChip – keep it simple

Client code (android app) - [here](#)

Client technologies and tools (android app):

- XML via Android Studio – design.
- Java – functionality and dynamic page/app-screen construction.
- NFC (android API) – to write and read from NFC tags.
- Android API to get access to phone action.
- HTTP request – to contact the server via Java objects and json (Gson – google API).

Client code (desktop app) - [here](#)

Client technologies and tools (desktop app):

- WIN FORM via Visual Studio – design.
- C# .NET – functionality and dynamic page/app-screen construction.
- HTTP request – to contact the server via C# objects and json (Gson – google API).