Bilkent University
Department of Computer Engineering

Senior Design Project

A-Tab: Artificial Tabldot

Final Report

Gürkan Gür, Arda Kaan Gültekin, Mahir Efe Macit, Subhan Ibrahimli

Supervisor: Halil Altay Güvenir

Final Report

April 30, 2021

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2

Table of Contents

1. Introduction

In today's world, time inefficiency is a common issue everyone faces. This problem can occur at any time in daily life. Anyone can face this problem while driving, working or even trying to eat outside. Although time efficiency is a common problem, the solutions of these difficulties are neither trivial nor common. To solve the time inefficiency, the root of the problem should be examined.

In cafeterias, restaurants and pubs, people buy meals through engaging with cashiers or sellers. The process of ordering and paying for meals can possibly be stressful and tiring when the cafeteria is overloaded. Reducing the time of ordering and paying processes can make a great save in terms of time efficiency.

With the A-Tab project, our aim is to reduce the time process of asking the cost of the meal. A-Tab aims to give the cost of the meal that is ordered by the customer without any interaction with the cashiers by scanning the meal. The project is significant in terms of the innovative tools that will be used through accomplishing its aims and can be improved in the future to accomplish other various tasks, like other image analysing applications.

In the Final Report we are going to go through our requirement details as both functional and nonfunctional, then we will thoroughly explain our design, development and testing details. We will talk about our plans and details of maintenance. Then we will be talking about ethics, judgements and teamwork details. Last but not least we will briefly talk about the new knowledge we learned through the process and the future work we can do.

2. Requirements Details

2.1. Functional Requirements

2.1.1 Food Detection

A-Tab should process the taken photo from the camera and identify the food objects. A-Tab should understand and determine the food correctly and give the price of it.

2.1.2 Server/Database Requirements

A-Tab should be connected to the server continuously to calculate the similarity of the scanned images and after that it should decide the kind of food number of foods in the plate. If the similarity ratio is high however, the program can not link the scanned object to the objects in the Database, it should take the object to the Database.

2.1.3 Setup Requirements

Setup should be appropriate for efficient use of A-Tab, both in terms of physical setup and hardware setup. As a physical setup camera should be placed in a way that all the foods in the tray could be seen from the camera. The brightness and the quality of the camera should be appropriate. In terms of hardware setup the programs and utilities that we will discuss should be installed properly. In terms of software setup, Flask, Flask_mail, PysimpleguiQT, utils, PIL, matplotlib, collections, io, zipfile, tensorflow, tarfile, sys, six.moves.urllib, os, pyside2, cv2, sqlite3 and numpy libraries and packages have to be installed. Python 3 has to be installed.

2.2. Non-Functional Requirements

2.2.1 Extensibility

This will be our main focus since our project can be developed and used in a significant number of places. Our view of handling will help the project to be developed in the later stages(after the project is done). There will always be a better or faster solution and the interface of a program can always be improved. So the project should be extensible.

2.3.2 Efficiency

A-Tab application should give the result of the taken image and process it in less than 20 seconds in order to be time efficient. Time spent in the restaurant line should be deduced in order to have a time efficient project. The communication between the database and the program should be minimal in terms of time to be a time efficient program.

2.3.3 Scalability

The A-Tab program is not just for a specific restaurant, so the database and system should work concurrently with many restaurants and the app should not be affected by the number of customers or the number of restaurants using it. It should handle multiple images from different users simultaneously.

2.3.4 Usability and Accessibility

The system should be user friendly. Although processing an image is a hard process. The program should be user-friendly and easily used by users. It should be clear when the program should interact with the user whether a photo could be processed or not. The application should be free however, to run the application the program will need an additional camera to take the photos of the food. It should be downloaded for the restaurants.

2.3.5 Error Handling

Errors and bugs should be noticed as soon as possible. Expected errors are identification problems and pricing problems. In case of identification errors unidentified objects should be added to the database. Pricing problems should be handled in the database also. Only way to overcome these errors is software changes, because we cannot show an error message especially for unexpected conditions. However, for identification, the object frame will appear on the console.

2.3.6 Maintainability

In terms of arrangements and changings in datasets for one restaurant, only the specified restaurants dataset should be changed. Moreover also the changes in the backup side of the project should not affect the whole algorithm to maintain the modularity of the project.

2.3.7 Reliability

The system should guarantee the privacy of the datasets of the restaurants and make the calculations correctly.Moreover there will be no access from interface to the server side to ensure the privacy of the data.

3. Final Architecture and Design Details

We implemented a system in an object oriented perspective. We followed the 3-tier design pattern. There are three main parts in the system. LayoutManager is responsible for the front end part of the system. DataController is responsible for the database system of the application. VisionAnalyzer is responsible for the logic part of the system. All controllers in the system are singleton classes. In MainController, all the system elements are combined.

3.1. GUI

We used a simple user login/register usage. It is not unique in the fact that it does not do anything different than some login/register systems. We added an authentication part in the register part to send the user's email address an authentication code.

3.2. Logic

We decided to process the frames from the camera by simply opening the camera through the application. The photos will not be taken by the user. User will only open the scanner. With that our machine learning/object detection algorithm will be invoked. While getting the frames from the webcam, our detection algorithm will work constantly, and continue until the recording is stopped by the user.

3.3. Database

Database part of the system will create a user.db database. Connections, creating a table, and functions like adding a user to the database system will be done through the database part. Sqlite3 library will be used in order to manage the database system.

4. Development / Implementation Details

4.1 Front End

PysimpleguiQT, which is an extended version of Pysimplegui library is used in the front-end of the application. LayoutManager class has all the layouts that are used in the system as private variables. We have separated the menus to different classes, and every class creates its

own layout. LayoutManager collects those layouts and combines them in the window. To make it more understandable, here are the private variables of the LayoutManager class:
These are classes which the LayoutManager holds one instance of every one of them

__startMenu
__registerMenu
__loginMenu
__userMenu
__cameraPage
__creditsPage
__settingsPage

These are the layouts of the menus and pages which LayoutManager collects from classes above and combines them in __allLayouts

__startLayout
__registerLayout
__confLayout
__loginLayout
__userLayout
__cameraLayout
__creditsLayout
__settingsLayout

__allLayouts

This is the window variable that all layouts are added into.
__window

LayoutManager creates a class instance and gets its layout as:

```
self.__startMenu = StartMenu()
self.__startLayout = self.__startMenu.createLayout()
```

Layouts are collected in __allLayouts as:

```
self.__allLayouts = [[sg.Column(self.__startLayout, key='-COL1-'),
sg.Column(self.__registerLayout, visible=False, key='-COL2-'),
sg.Column(self.__confLayout, visible=False, key='-COL3-'),
sg.Column(self.__loginLayout, visible=False, key='-COL4-'),
sg.Column(self.__userLayout, visible=False, key='-COL5-'),
sg.Column(self.__cameraLayout, visible=False, key='-COL6-' ),
sg.Column(creditsLayout, visible=False, key='-COL7-' ),
sg.Column(settingsLayout, visible=False, key='-COL8-' )]]
```

Event loop in the LayoutManager changes layouts and performs actions according to mouse and keyboard events.

Event loop is a standard while loop, and interactions are done like:

```
while True:
    event, values = self.__window.read(timeout=20)
    if event == "Exit" or event == "Exit0" or event == sg.WIN_CLOSED:
      break
    if event == 'Register':
      self.__window['-COL1-'].update(visible=False)
      self.__window['-COL2-'].update(visible=True)
      …
```

Different from other pages, the camera page uses cv2 and numpy libraries.
Outside of the event loop, two variables are created: cap and recording.
Cap variable takes video capture function of cv2 as:
Cap = cv2.VideoCapture(0)
Recording variable is a standard boolean variable which shows if the camera is recording right now. It is set to False initially.
In the event loop, these variables are used to capture motion as:

```
    if event == 'Record':
      recording = True

    if event == 'Stop':
      recording = False
      img = np.full((480, 640), 255)
      imgbytes = cv2.imencode('.png', img)[1].tobytes()
      self.__window['image'].update(data=imgbytes)

    if recording:
      ret, frame = cap.read()
      imgbytes = cv2.imencode('.png', frame)[1].tobytes()  # ditto
      self.__window['image'].update(data=imgbytes)
```

While recording, the logic part of the system analyses the motions through VisionAnalyzer.

4.2 Database

DataController class is responsible for the database implementation of the system. We have used sqlite3 library to create the database system. Flask and Flask_mail libraries are used to send a confirmation mail to a desired mail address. DataController like the other controllers is a singleton class which means only one instance of the class can be created. This instance is created in the main controller.
There are 7 functions in the DataController as:

1 - def userData(self):
This function connects to the database and creates an empty table for the users as:

```
con = sqlite3.connect("user.db")
    cur = con.cursor()
    cur.execute("CREATE TABLE IF NOT EXISTS  user (id INTEGER PRIMARY KEY,
RestName text, Email text, Password text, Phone text, Address text, Foods food)")
    con.commit()
    con.close()
```

The sql statement, "CREATE TABLE IF NOT EXISTS user (id INTEGER PRIMARY KEY, RestName text, Email text, Password text, Phone text, Address text)", creates the table for the desired inputs.

2 - def addUser(self, RestName, Email, Password, Phone, Address):
This function adds user to the table with the given variables.
The sql statement used in this function is ("INSERT INTO user VALUES (NULL, ?,?,?,?,?)", (RestName, Email, Password, Phone, Address))

3 - def viewData(self):
This function returns all the datas on table as a row, which is done by the sql statement "SELECT * FROM user" .
In the returned row, all the user data are accessible. To get a desired user data, an algorithm like this is used:

```
for x in datacontroller.viewData():
  if x[2] == values['-regemail-']:
    loginUser = x;
```

In this algorithm, the user which is found by the given -regemail- variable is copied into loginUser variable.

4 - def deleteRec(self):
This function deletes user record from the database with the statement ("DELETE FROM user WHERE id=?", (id,))

5 - def setMail(self, rec):
This function creates a confirmation code for the given email which is then later sent to the email address in another function.

```
self.recMail = rec
self.__regCode = random.randint(10000, 100000)
return self.__regCode
```

6 -def sendMail(self):
This function sends confirmation email which is necessary for the registering process. It is done as:

```
with app.app_context():
  msg = Message(subject="Confirmation code for A-Tab",
        sender=app.config.get('MAIL_USERNAME'),
        recipients=[str(self.__recMail)],
        body=str(self.__regCode))
  mail.send(msg)
```

7 - def changeUserData(self, key, changeElement, newElement):
This function changes the data of the desired user.
In function parameters, key implies which user needs to be changed, changeElement selects which data to change(email, password, vs) and newElement is the element that will take the place of the old one.

4.3 Logic

To detect food objects we used pre-trained Convolutional Neural Networks(faster_rcnn_inception_v2_coco_2018_01_28) to extract the features from the images of our classes. We mostly used the library of TensorFlow for both detecting the object and training our model. We used boxes to indicate the food that we have found. Rather than taking the photo of the “action” scene we used the webcam to capture the image live and detect the objects while the webcam is getting the correct scene.
Logic actions are done on the class ‘Vision Analyzer’.
At first glance we planned two different things from what we have implemented. We planned to run the object detection on the server but since our main concern was time efficiency and its response time from the server could be an issue we eliminated that option. Also we planned to add a feature to have a more responsive program to let customers train their own foods by adding the images of the foods. But initially we saw that customers will have to take the same picture over and over again, and the loss would be higher than we want. As a result the training/accuracy would be low. So we decided to have a predefined dataset that we feed the model and the customers select their food from the dataset. Also while creating the dataset we manually labeled the food photos to feed our model into.

5. Testing Details

5.1. Object Detection

Our application's accuracy should be higher in order to have an efficient program. In order to get a high efficiency we needed to have relatively small loss values. Since the lower values of loss indicates the efficiency of the trained model. So to have a significantly lower loss value we tried to train our model with big sized datasets which includes the images of the foods that we want to integrate in our model. We used "Label Image" to process our datasets to be able to start feeding our model. In the Label Image program we manually selected the corresponding food object and labeled them by their names.

5.2. Camera

There will not be any interaction with the camera and the user other than the scan button itself. We wanted the interaction between the camera and user to be as minimal as possible in terms of user view. However, they might need to adjust their cameras according to brightness and the location of the object. So to test the "camera" we simply run our application in different types of cameras to see whether there are any complications related to camera quality. We didn't observe any big problem.

5.3 User Interface and Database

We have tested the user interface through performing all the actions and their combinations. Database is observed through actions and no problem or inaccuracy found.

6. Maintenance Plan and Details

Since we do not have any complications on our database end we only store simple credentials of the users. The only maintenance we might need to do is when the customers want to add food that is not on the dataset. We need to label the new dataset and add to the system so the users can do it.

7. Other Project Elements

7.1 Consideration of Various Factors in Engineering Design

As usual, developers are supposed to design software development plans in order to determine tasks to work on within the development. Our approach is that if the plan is considerable and applicable, then fundamentals of a good software development are

successfully settled. That is the reason why we have been putting an effort to consider carefully and seriously not to face serious problems in the future.

In the process of composing a software development plan, some essential factors are taken into consideration. Following the software engineering theory, we examine that main factors that should be considered throughout the development plan are the type and size of software, the experience of predecessors, the degree of complication of obtaining users' needs, in which the user could be either customer or cashier, the development techniques and tools, the condition of development team, development risks.

The first factor we put into the consideration is the type and size of software since it plays a significant role. As we are planning to make and use a specific database for the software development, different size and type of data, their particular architectures, their amount of data store, their pattern processing data, their operating environment, their serving object and etc. should be considered beforehand so that we can reasonably design software development plans.

The second factor we put into the consideration is the experience of predecessors to enhance efficiency and quality of the software. After the assurance of the first factor, we should examine the similar softwares if there are any available, so that we can conclude the advantages and disadvantages of each software and bring a better and more convenient software development plan for our project.

The third factor we put into the consideration is the degree of complication of obtaining users' needs, in order to satisfy users' requirements and desires. Throughout the investigation we need to be confident about understanding some details in users' needs, which could be different scenarios, such as determination of stability and clarity in users' needs. In case of uncertainty, instability etc., developers need to put an effort and be confident that those points are fixed.

The fourth factor we put into the consideration is the development techniques and tools, in which developers should clearly identify which techniques and tools can be convenient for execution in the development. Because this factor is included in essential phases, which are system analysis phase, system design phase and implementation phase. Therefore, developers

should be careful in choosing development techniques and tools, and should make a proposal of using certain techniques and tools, and clarify the reason behind that.

The fifth factor we put into the consideration is the development team, which has an inevitable role in the software development plan. Because if there is no passion to work on the development in each team member, there is a high possibility that the project will not progress as it is desired, no matter how good the software development is. Therefore, the strength, condition and thought of each developer in the team should be well-known to the team in the beginning phase of the development.

The sixth factor we put into the consideration is the development risk, which frequently occurs within the development. There are some methods used during the software development for predicting future risks, their types, possible solutions and the outcomes from the solution. On the analysis phase, one of the main tasks is to determine problem definition, feasibility and needs analysis. In every stage of the development, there are multiple different types of risks that could appear. Developers need to know and predict possible solutions and be aware of the difficulty and probability of being able to solve the problem, and by making use of those data, they should be able to conclude possible outcomes and be ready for different scenarios.

7.2 Ethics and Professional Responsibilities

Ethics and professional responsibilities are two of the main important parts that we had to be careful about. In order to make everything work out for us, after the implementation we are going to test in a cafe environment and make a kind of simulation before applying to the real situations. We are aware that there will not be any threats to the users' data since we considered mandatory measures in the application. Within the time, we will update the security system if needed.

7.3 Judgements and Impacts to Various Contexts

In the development process of A-Tab, we have also observed that application would have various effects on other different topics. The factors such as public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors have effects

on software engineering design. We evaluated these factors and discussed how they would affect the design. Detailed information was discussed in the analysis report as well.

	Effect Level	Effect
Public Health	8	It will reduce the contact between people since it will reduce the time spent in queue
Public Safety	7	Private information of the users will not be distributed to any third party company
Public Welfare	2	Appropriate pricing
Global Factors	2	Applicable in any country
Cultural Factors	4	Culture Based sets are used
Environmental Factors	0	None
Economic Factors	5	Increasing the number of sales by reducing the queue length and time
Social Factors	4	Can be used by anyone in terms of age,country

Figure 1. Evaluation of factors in engineering design

7.4 Teamwork Details

In software projects, human and social factors play indispensable roles. The management of knowledge and creativity are essential for software development, and creative work connects collaboration and interaction. That is why teamwork is considered one of the significant skills

for the development process. In a project development, we are trying to contribute and fully understand every part of the project and share the information with other developers in the meeting and update the information that way in their mind. Gürkan Gür has been mainly responsible for the user interface and database system of the project. Mahir Efe Macit, Arda Kaan Gültekin, and Subhan Ibrahimli worked on vision analyzing system of the project.

7.4.1 Contributing and functioning effectively on the team

The ability of each developer to contribute within a team may successfully manage the execution of software development. Throughout the development of our project, we have arranged online meetings, analysed the improvement, discussed the problems and details of the project, separated different parts and distributed them to each developer to work on.

7.4.2 Helping creating a collaborative and inclusive environment

In order to make a collaborative and inclusive environment, we have been trying to create a friendly environment for a smoother and looser structure, motivate the individuals that are engaged in the development and create a healthy surrounding, encourage multi-disciplinary work, actively involve all project members in collaboration and etc. We have also tried to identify each developer's strengths and determine what contributions that developer could make. This way, one member could compensate for the disabilities of others, and the appropriate tasks are given to each developer according to their strengths.

7.4.3 Taking lead role and sharing leadership on the team

In general, the type of software engineering team that is made in our environment could usually consist of four to six people. Each member in the team is responsible for a particular set of capabilities. In our team, the leader role does not imply being expert on something, but instead it is a role change. Team leader is responsible for controlling the team's execution and the quality developers produce, and overall growth of the individuals on it. We do not have a consistent team leader in our development environment since the team size consists of a small number of people, and we can easily follow each developer's improvement on the project, the progress they make, and their overall growth throughout the development.

7.5 New Knowledge Acquired and Applied

As many tasks in software development involve the acquisition of knowledge, our team also had to do research about object detection with web cameras and images, in order to execute the desired functionalities for our application. Before the implementation, most of the team members had no much knowledge or experience with them. Therefore, by implementing our application we have gained some experience and acquired knowledge, then applied for the project.

There were various tutorials and researches about object detection with TensorFlow. Therefore, We have chosen a few of them and tried to set up the environment and tools, and learn how to use them for the development of our application.

8. Conclusion and Future Work

8.1 Conclusion

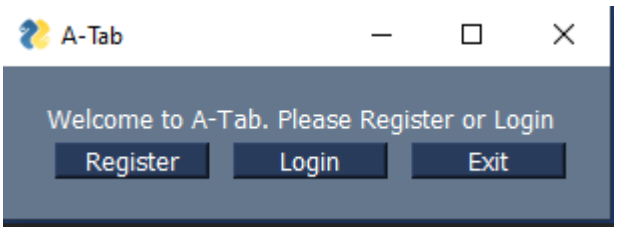
We have implemented an application under the name of A-Tab project, aiming to reduce the time process of asking the cost of the meal. A-Tab aims to give the cost of the meal that is ordered by the customer without any interaction with the cashiers by scanning the meal. The main goal was to make the procedure as much comfortable as possible for both customers and employees in the cafes. We believe that the application we are going to provide will be useful and desirable for most customers since we are aware of the value of the time for them. We hope this application will be beneficial for the most places for its usage.

8.2 Future Work

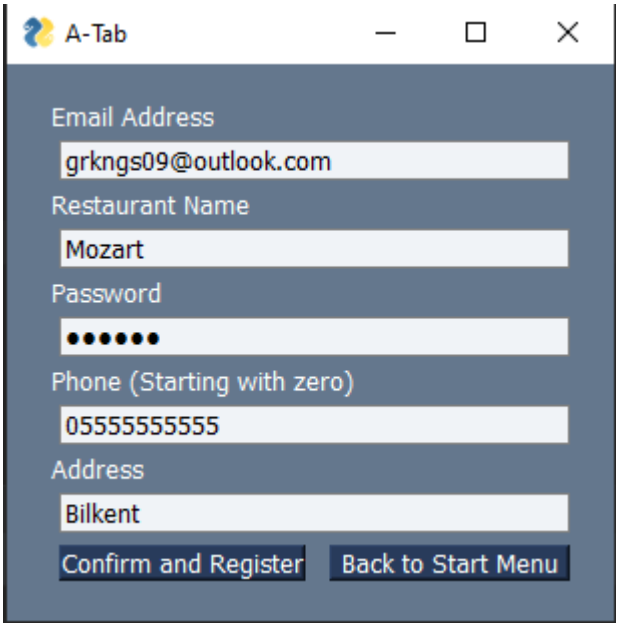
The first release is going to be examined in case any cafes are going to agree to apply. This way, we are going to observe and get any other suggestions for the future work as we are going to get feedback from the users. In case of any other good suggestion to add for the application, we can update accordingly. Other than that, we will try to enhance the efficiency and the speed as much as we can within the process.

9. User Manual

A-Tab has a very simple user interface designed for the ease of use.
Application starts by running the main script file main.py.
After running the file start menu pops up:

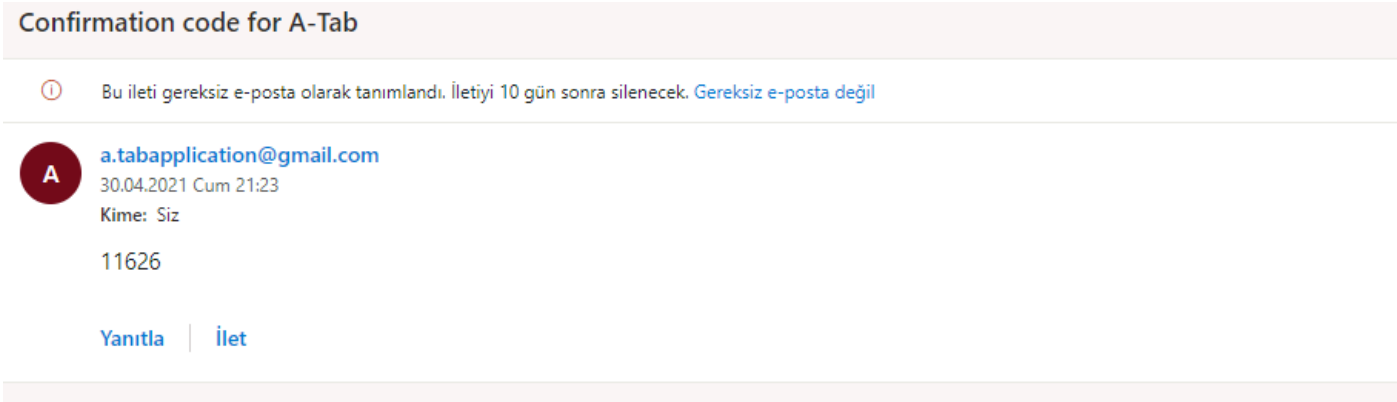


Clicking on the register button will open the registration menu:

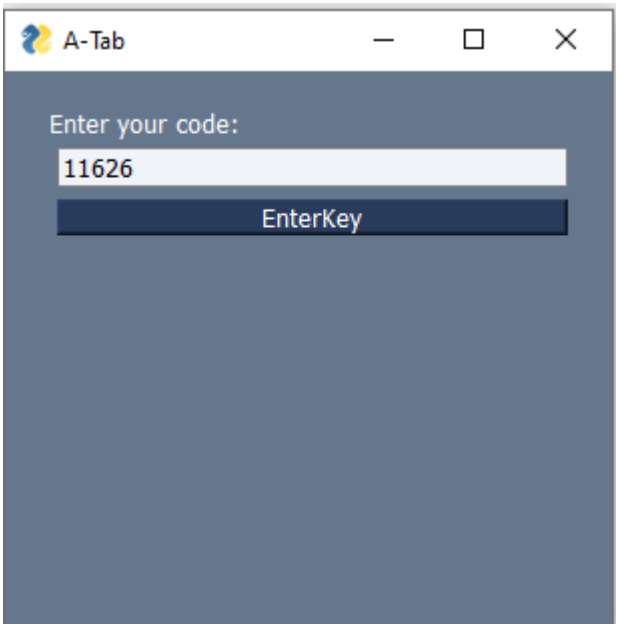


In the registration menu, all the boxes have to be filled in order to register.
User has to enter a valid email address.
Phone number length has to be 11.

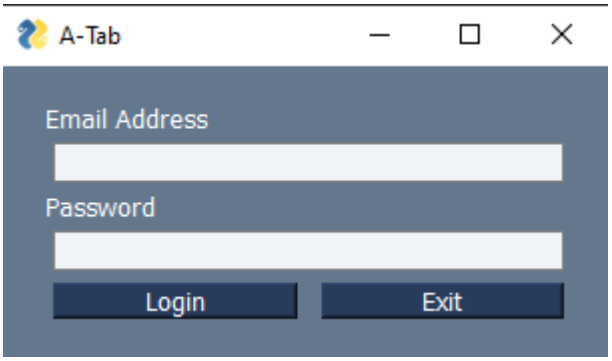
After clicking ‘Confirm and Register’, a verification code is sent to the written email address,(which may drop into spams or unnecessary folders) such as:



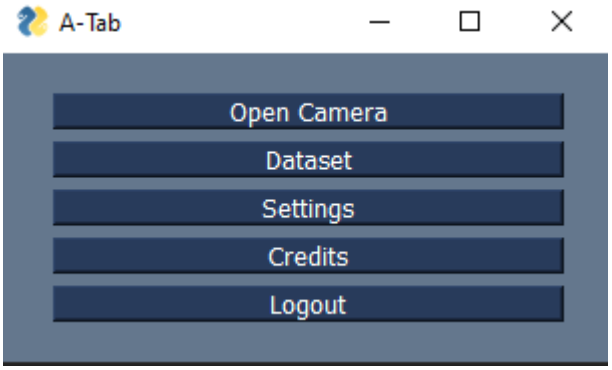
Entering this code to the box will finish the registration process:



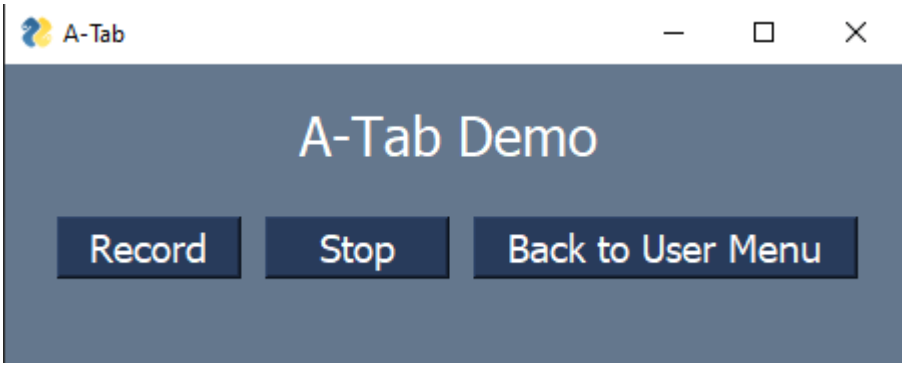
To login, all the user has to do is click on the login button in the main menu. After that login menu opens up:



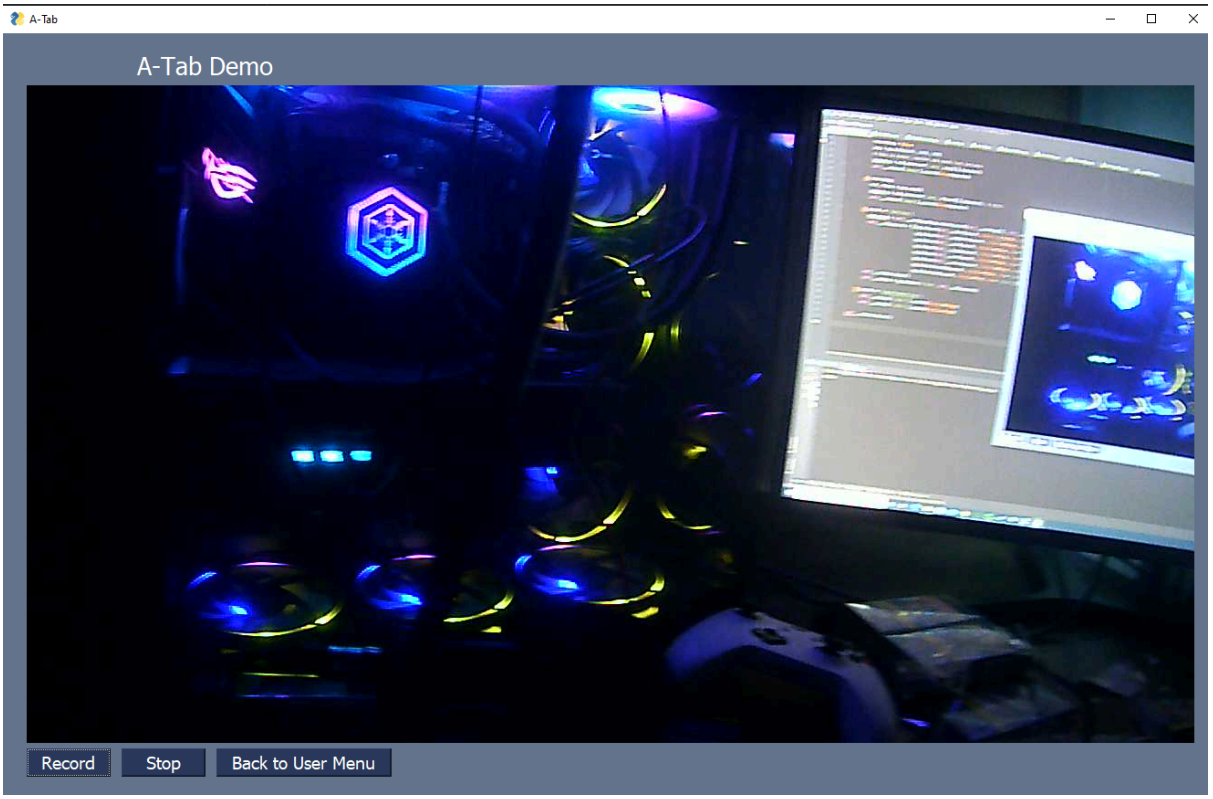
Writing correct credentials will open the user's menu:



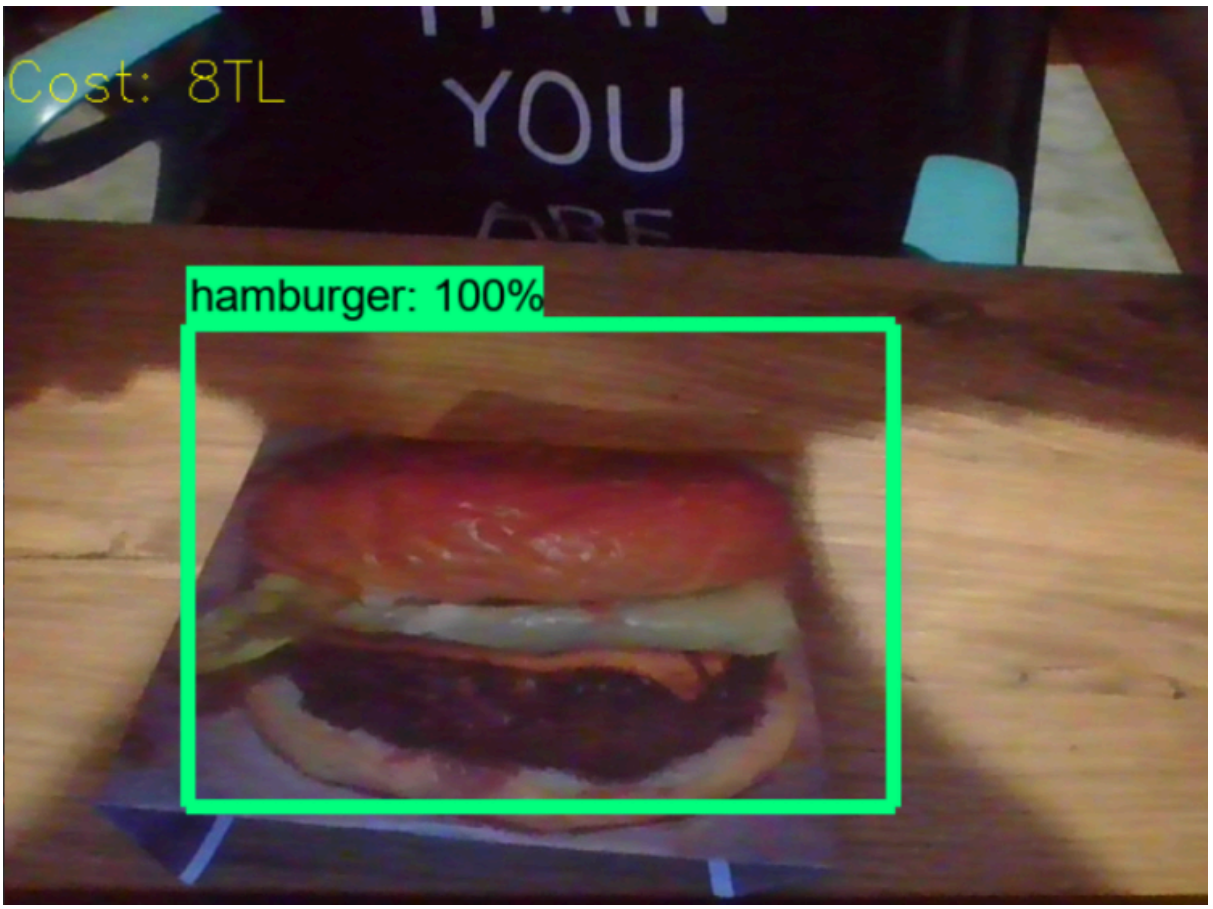
Currently, users can not make any changes on the dataset or food prices. Dataset and food prices are determined by the A-Tab developers, and entered to the desired user's database.
Clicking on 'Open Camera' will open the camera page:



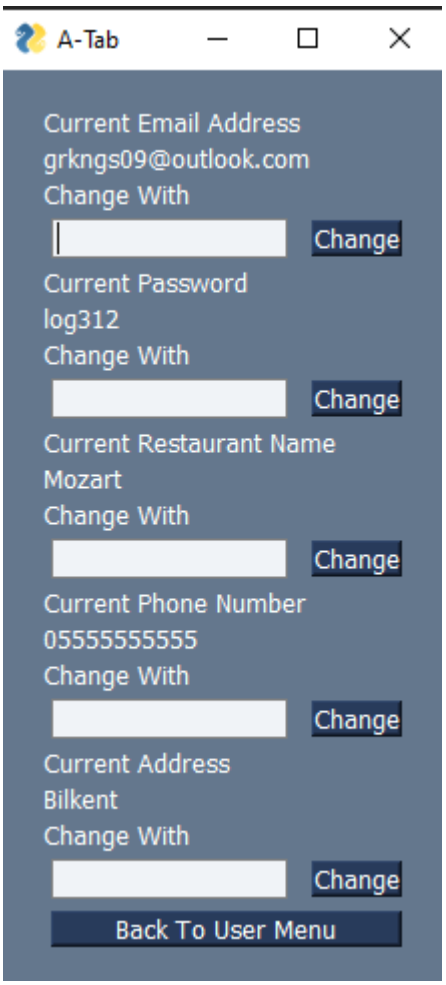
Clicking on record will start the camera which has been connected to the computer. If there isn't any camera input, the application will stop working.



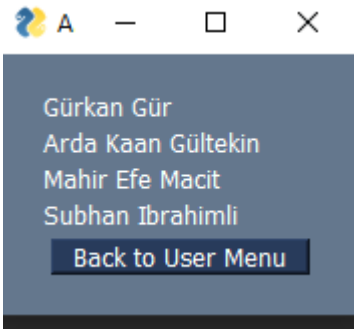
While showing the foods which user needs to see the price of it to the camera, screen will write the food's name, price, and accuracy percentage:



Clicking on ‘Back to User Menu’ will open the user menu again.
Through clicking settings, user can change the credentials:



Credits Button will open the credits menu which shows the developers:



And the Logout button will perform the logout process.