SmartMeal: A scalable catering application

Mohamed-Ata-Radu Abu Ras, Paul Stefan Popescu, Mihai Mocanu, Marian Cristian Mihaescu

University of Craiova
raduradu0410@yahoo.com, {stefan.popescu, mihai.mocanu, cristian.mihaescu}@edu.ucv.ro

Abstract. In this paper, we present the design and development and a demo of SmartMeal, an application designed for an I.T. company that allows employees to order their meals for lunch. The application's audience is not limited and can be easily used by any company that regularly demands meals. This application uses a mix of frameworks, programming languages, and technologies to provide all the required functionalities and run on desktop computers and mobile devices. Another benefit of this application is that it was designed and implemented to be scalable for more users. The interface of the application was designed to be easy to use and let users order meals fast and efficiently.

Keywords: User interface, User Experience, Application design

DOI: 10.37789/ijusi.2022.15.1.2

1. Introduction

This paper presents a project designed for an I.T. company where employees can order the desired meal for lunch. The application was designed to be easy to use and offer a great user experience. The project comprises two main components: the desktop application and a mobile solution that aims to facilitate system administrators' usage. Once the application finished development, it was launched on the market, and the first big problem appeared: the increased number of users which made plenty of traffic and usage around lunchtime. An interesting approach was adopted for this problem, using PWA (progressive web application) Magomadov (2020) for the desktop module.

One of the main challenges of software applications these days is scalability. Because of the increased mobile traffic, several approaches for backend performance optimizations were adopted, among which is the concept of local data storage. This concept brings two main benefits: the data is accessed faster because the necessity of the internet connection is eliminated, and the database is less queried, which removes the crash possibilities.

For this approach, the development environment is composed of C++, C#, and Python, along with Qt framework, QML, and SQL used in Qt creator, visual studio, and Android Studio. In this context, Python was used for creating a testing platform because this language offers plenty of modules developed for this purpose. Qt was used in the application because it is a set of free and open-source widgets which runs on different software and hardware platforms. The most important reason for choosing Qt is that it can be used with different compilators (in this case, CGG C++ and Visual studio suite) and includes Qt Quick, which includes a scripting language called QML that allows the usage of JavaScript for business logic.

To develop the mobile/desktop application in the same project, we used different tools along with qMake, which allowed us to describe compilation processes for other platforms and environments.

Qt Creator is a development environment that offers design and development tools using the Qt suite. Qt creator was used in this situation for project compilation and to obtain a .apk version for mobile devices. Qt creator also allows users to create installation packages for mobile devices suitable for publishing in stores.

One of the most important modules of the application is PWA integration which enables new capabilities for modern browsers Tandel & Jamadar (2018). Using service workers and a web application manifest, the web application may become reliable and usable. If there are no new capabilities, the users will continue to use the essential experience. Regarding the numbers, the companies which launched PWA applications had significant improvements; for example, Twitter had a 65% increase in the number of pages per session and 75% more tweets with a 20% less rejection rate, and all of these came with a 97% reduction of the application dimension. After Going to PWA, Nikkei had a 2,3 times increase in organic traffic, 58% more subscriptions, and 49% more daily active users. Another good example is Hulu which replaced its desktop experience with a Progressive Web Application and got a 27% increase in returning visitors.

To implement PWA, we used Qt's framework abilities to save offline data in JSON format in its dedicated folder, which is dependent on the operating system (for example, in Windows is C:\Users\User \AppData\Roaming).

Python stress test Lei, et. al. (2014) was used to simulate a large amount of data (in this case, users) to evaluate the results obtained after PWA was implemented into the application.

2. Related work

Regarding meal and nutritional apps, there is research related to software engineering and medicine. In López et. al. (2017), a smartphone nutritional application for making smart and healthy choices when buying food in grocery stores is presented and tested for its feasibility, usability, satisfaction, and acceptability, so it is also HCI related. The application's goals were to improve food selection when purchasing foods in grocery stores based on a pre-defined budget, to improve dietary patterns based on the Dietary Guidelines for Americans Mosher et. al. (2016), and to improve weight status. It was evaluated within a randomized pilot trial. Another but the more health-related application is presented in Ambrosini et. al. (2018), which analyzes the feasibility of a commercial smartphone application for dietary assessment. They tested the acceptability and relative validity of a commercial smartphone application as an epidemiological nutritional assessment tool.

Another HCI and nutrition-related paper Robinson (2013) presents the development and feasibility study of a smartphone-based attentive eating intervention. They focus more on nutritional facts to encourage individuals to eat more attentively to help reduce calorie intake.

This idea of Smart Meal can go even further in a context of a smart city for intelligent living as described in Liyanage et. al. (2013), which explains the thoughtful planning for SmartMeal and the arrangement of related services to succeed in the approach. To increase their approach's accuracy and efficiency, they also developed a questionnaire and used its results for better understanding and improvements. But this type of application may be relevant not only for a company as it is presented in Ibrahim (2017) where the authors propose an application for school canteen: Meal-Go Application. Their project aimed to develop an application system for school canteen purchases. Their task was designed to improve the traditional or current method used in managing students' school meals. The same idea was also addressed in Blaas, R. M. (2019), which refers to the problem in the University context.

Another related and exciting paper on meal ordering is Attard (2020), which uses heuristic methods for solving a meal delivery routing problem. In this paper Attard (2020), two constructive heuristics are used: the Nearest Neighbour (NN) and the Parallel Savings (P.S.) to generate initial routes, which are subsequently improved using Local Search improvement heuristics. Since the Local Search method tends to get stuck in local optima, the Globsl Local Search meta-heuristic is applied to the Local Search routes to improve them further. Results show that the P.S. solutions are better than the N.N. solutions and that the local search method converges to deep local optima.

One last relevant paper which describes a related application is Xiaowei, (2015), which presents a hotel intelligent Android platform-based ordering system. At the functional level, based on object-oriented design, it fully integrates hotels and meal business needs. The method of the client module includes dinner, dishes search, user reviews, user login function, client-designed menu management, order management, system management, and appraisal management.

3. System design

The project also respects the MVC architecture (Model-View-Controller), an architectural model that separates the application's logical components. The model component corresponds to the whole data logic that the user uses. For example, in our case, a *Customer* object will take the information about the client from the database, do some work, and then update them in the database. The view component is used for the whole U.I. of the application. For example, Client visualization includes all the U.I. components like text boxes, dropdowns, etc. The Controllers act like an interface between the Model and View for processing the whole business logic. For example, in our case, the Customer controller will manage all the interactions and inputs from Customer View and will update the database using Customer Model

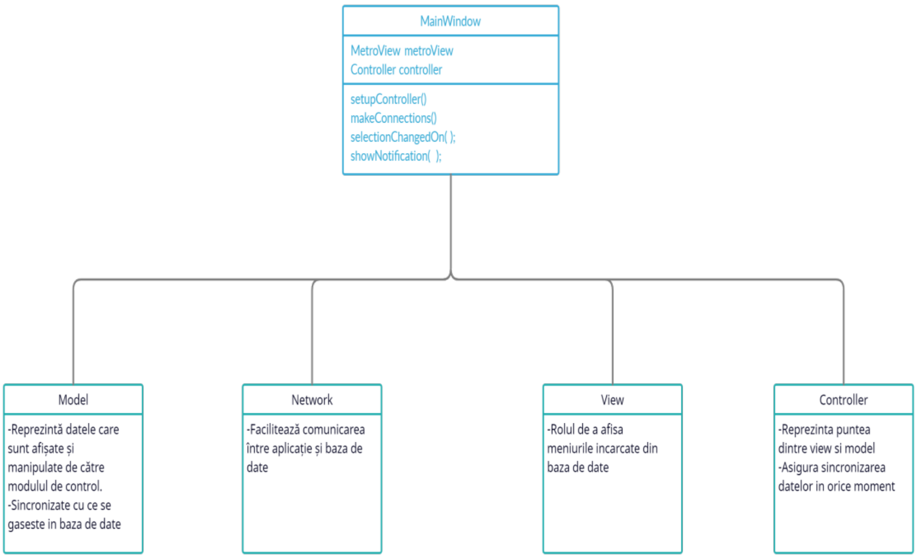

MainWindow
MetroView metroView
Controller controller
setupController()
makeConnections()
selectionChangedOn( );
showNotification( );
Model
-Reprezintă datele care sunt afișate și manipulate de către modulul de control.
-Sincronizate cu ce se gaseste in baza de date
Network
-Facilitează comunicarea între aplicație și baza de date
View
-Rolul de a afisa meniurile incarcate din baza de date
Controller
-Reprezinta puntea dintre view si model
-Asigura sincronizarea datelor in orice moment

Figure 4. Main System Architecture

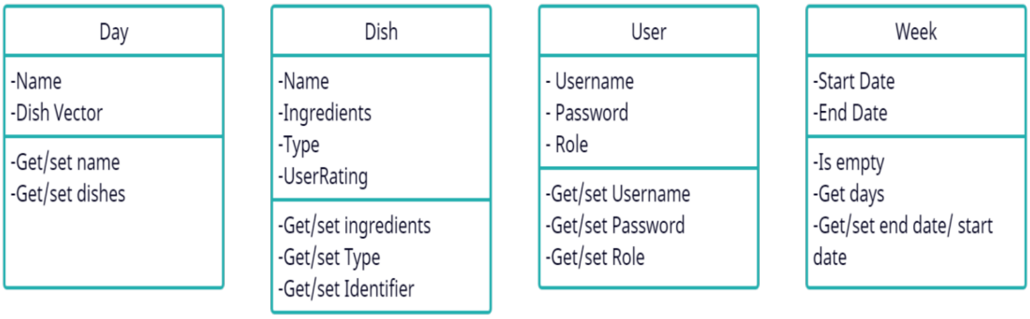


Figure 5. Model architecture

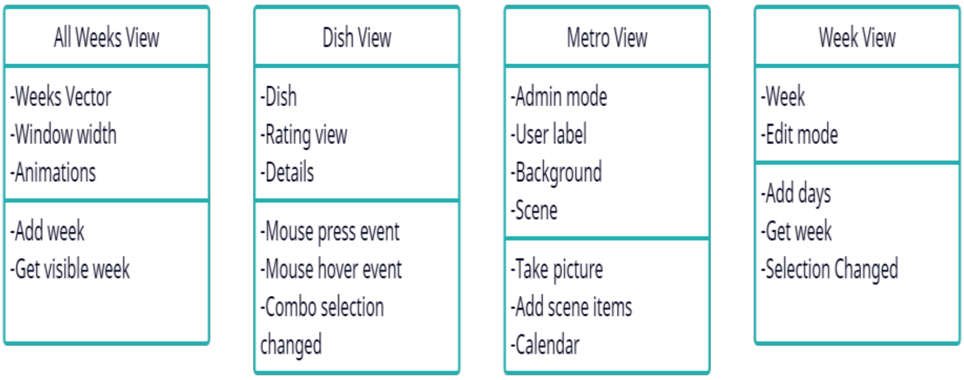
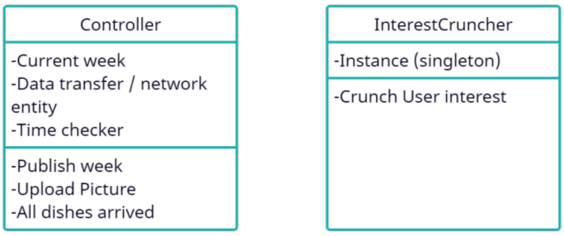


Figure 6. The architecture of the View Package

Figure 1 presents the main modules of the application concerning the MVC architecture. On the top of the figure, we have the main window class, which connects the Model, the view, and the controller, and a network module that facilitates the connection between the database and the application.

Figure 2 Presents the classes from the Model, which are Day, Dish, User, and Week, each having getters and setters for their attributes. These classes are used to load the data from the database.

Figure 3 presents the classes from the View package; these classes use the ones from the module and also implement several actions for viewing the dish (in Dish View), for taking pics and adding items (in Metro View), and for viewing the menu by week(s) (in Week View and All Weeks View). Each of these classes has several attributes used for the actions implemented by the methods from these classes. Still, most of the business logic is implemented in the following figure, which represents the controller package.



Controller

-Current week
-Data transfer / network entity
-Time checker

-Publish week
-Upload Picture
-All dishes arrived

InterestCruncher

-Instance (singleton)

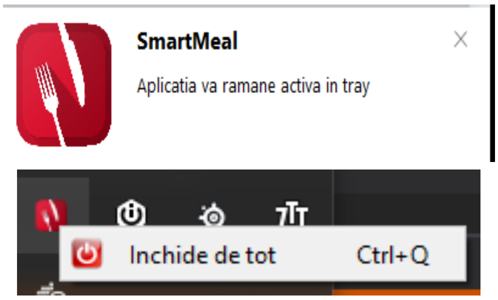
-Crunch User interest

Figure 7. Controller Package

Figure 4 presents the Controller package, composed of two components inside this application: the classic controller, which uses the data and communicates with the database, and the "InterestCruncher" class, which analyses the ratings given by the users to the food dishes.

4. Application demo

The desktop application uses all the Q.T. framework's functionalities because it aims to obtain a fast and responsive U.I. It was designed to stay in the system tray, which is one of the user interface's most essential and used components.



SmartMeal

Aplicatia va ramane activa in tray

Figure 8. Application in tray

Figure 5 shows a snip from the application when minimized to tray, and we can keep the application in the system's try or close it permanently.

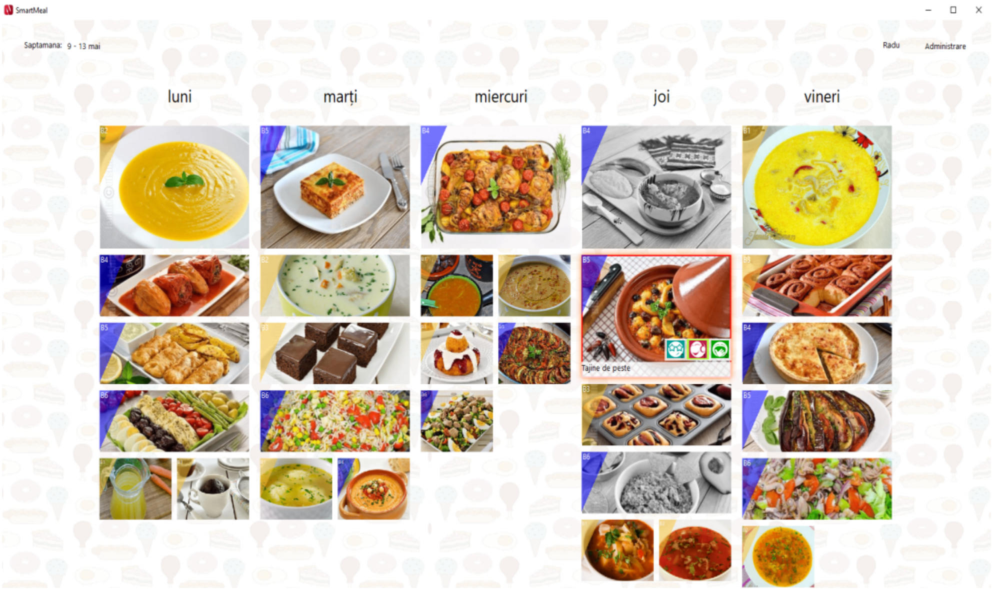


Figure 9. Whole week menu example

The application's main page is presented in Figure 6, which shows the menu for the whole week. From left to right, we can see the weekdays, and for each of these weeks, we can see the chosen food, but we can also select the desired week. For each day of the week, there is a list of available dishes as the administrators configured it. The end users can choose between the general words, but it's limited to only one option for each dish. One thing that needs to be mentioned is that each time a user chooses a plate, the U.I. is automatically updated. The application's main page is presented in Figure 6, which shows the menu for the whole week. From left to right, we can see the weekdays, and for each of these weeks, we can see the chosen food, but we can also select the desired week. For each day of the week, there is a list of available dishes as the administrators configured it. The end users can choose between the available dishes, but it's limited to only one option for each dish. One thing that needs to be mentioned is that each time a user chooses a plate, the U.I. is automatically updated.

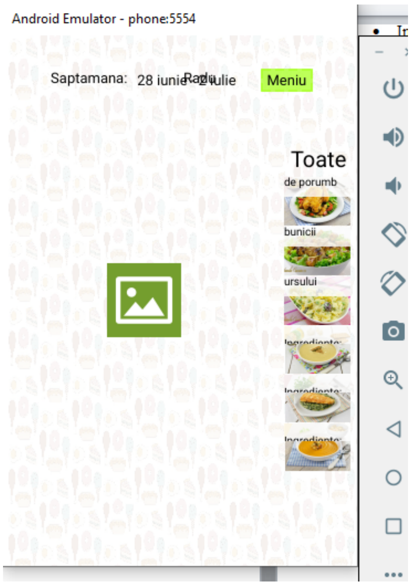


Figure 10. Android application capture

Figure 7 presents a short example regarding the mobile application. This part of the system was implemented to facilitate the image upload by the administrators when they create the menu for a specific day.

5. Conclusions

This paper presented a scalable SmartMeal application specifically designed for an I.T. company. The area of the article makes it relevant for the HCI domain as it tries to optimize the user interface and improves the interaction between the system. It's users by making it responsive even in high load conditions. It was designed to allow users to create their menu for each day of the week and to manage it easily.

In future work, a system evaluation shall be performed to better understand the users' needs and punctually improve the interface.

References

Ambrosini, G. L., Hurworth, M., Giglia, R., Trapp, G., & Strauss, P. (2018). Feasibility of a commercial smartphone application for dietary assessment in epidemiological research and comparison with 24-h dietary recalls. *Nutrition journal*, 17(1), 1-10.

Attard, K. (2020). *Application of heuristic methods for solving a meal delivery routing problem* (Bachelor's thesis, University of Malta).

Blaas, R. M. (2019). *Recipe and Meal management mobile application for Willieanne. nl* (Bachelor's thesis, University of Twente).

Ibrahim, I. B. (2017) Cashless Meal Application for School Canteen: Meal-Go Application. *CARNIVAL ON e-LEARNING (IUCEL)* 2017, 233.

Lei, K., Ma, Y., & Tan, Z. (2014, December). Performance comparison and evaluation of web development technologies in php, python, and node. js. In *2014 IEEE 17th international conference on computational science and engineering* (pp. 661-668). IEEE.

Liyanage, C. P., & Marasinghe, A. (2013, July). Planning smart meal in a smart city for a smart living. In *2013 international conference on biometrics and Kansei engineering* (pp. 166-171). IEEE.

López, D., Torres, M., Vélez, J., Grullon, J., Negrón, E., Pérez, C. M., & Palacios, C. (2017). Development and evaluation of a nutritional smartphone application for making smart and healthy choices in grocery shopping. *Healthcare Informatics Research*, 23(1), 16-24.

Magomadov, V. S. (2020, November). Exploring the role of progressive web applications in modern web development. *Journal of Physics: Conference Series* (Vol. 1679, No. 2, p. 022043). IOP Publishing.

Mosher, A. L., Piercy, K. L., Webber, B. J., Goodwin, S. K., Casavale, K. O., & Olson, R. D. (2016). Dietary guidelines for americans: implications for primary care providers. *American journal of lifestyle medicine*, 10(1), 23-35.

Robinson, E., Higgs, S., Daley, A. J., Jolly, K., Lycett, D., Lewis, A., & Aveyard, P. (2013). Development and feasibility testing of a smart phone based attentive eating intervention. *BMC public health*, 13(1), 1-7.

Tandel, S., & Jamadar, A. (2018). Impact of progressive web apps on web app development. *International Journal of Innovative Research in Science, Engineering and Technology*, 7(9), 9439-9444.

Xiaowei, D. (2015). The design of a hotel's mobile meal system based on the Andriod terminal. *Automation & Instrumentation*, p.12