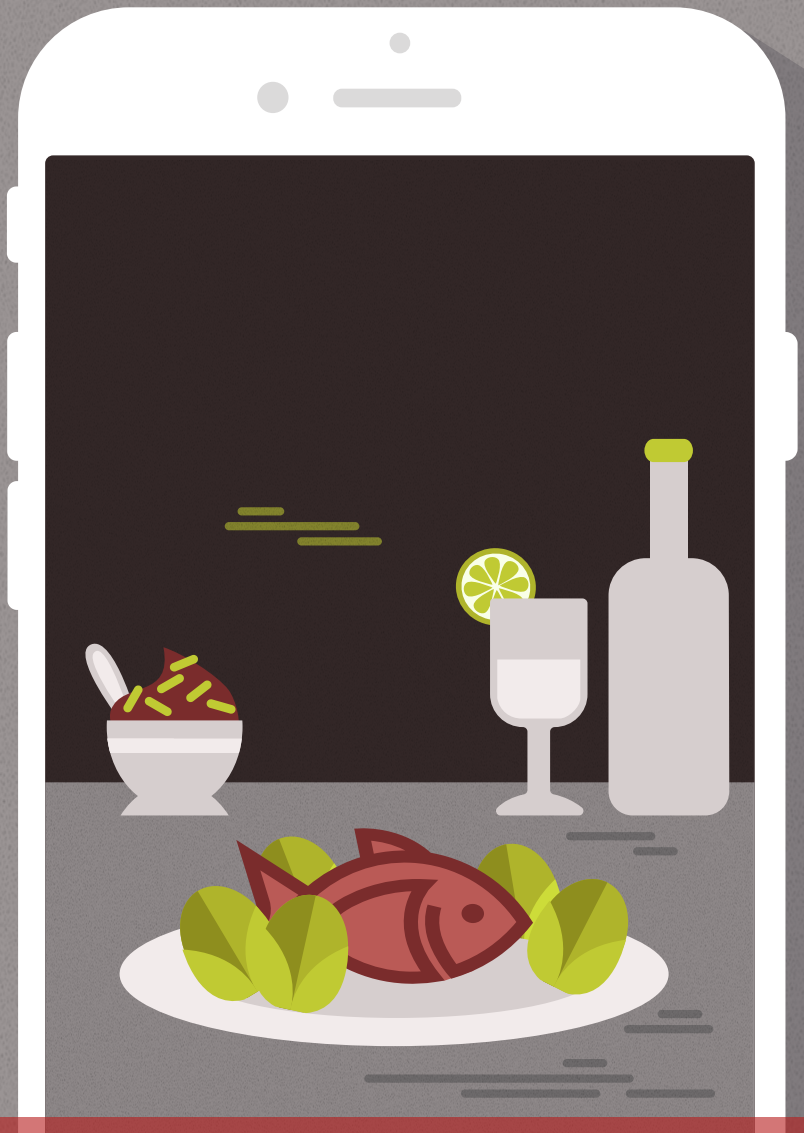


MOBILE ORDERING AND LOYALTY SERVICE FOR RESTAURANTS, CAFES, BARS

iOS, Android & Smartwatch, Web

Mobiversal

www.mobiversal.com

OVERVIEW

Mobile ordering and mobile loyalty programs are having a great impact on the restaurant business. “Mobile order and pay is enabling us to serve more customers more quickly and efficiently and to significantly reduce attrition off the line,” Starbucks CEO, Howard Schultz told analysts this summer. The “My Starbucks Rewards” loyalty program now has 10.4 million active members, up 28% from a year ago, and those shoppers now account for about 30% of business in North America. Mobile payments account for 20% of all in-store transactions in the U.S., more than double the figure Starbucks reported two years ago.

Key benefits of a mobile ordering and loyalty app:

- Increase number orders and bookings, by making it easier for clients to purchase
- Allows customers to order in advance or while they are on the move
- Keep existing customers and earn new ones, by providing a positive experience and offering rewards
- Improve internal processes and productivity for restaurant employees

We’ve compiled this proposal for a service that would help even smaller businesses to get access to a system that increases the safety, efficiency, and productivity throughout their operations. This is a full-featured platform where restaurants can signup and start benefit from the advantages of mobile apps.

ABOUT MOBIVERSAL

Mobiversal is a mobile app development company working with startups and established brands, like Forbes, to create unique mobile experiences. Since its beginning in 2011, Mobiversal has been focused solely on developing iOS and Android apps. The company was ranked among Europe’s "Top Mobile App Developers" (Clutch.co) and has turned ideas into apps for over 60 clients from more than 15 countries.

We work together with our clients to build compelling apps that provide lasting excitement and value to their customers. Our understanding of mobile and web based technology empowers us to bring a level of expertise to your company like no one else. We know how to build products that can scale to your business' needs, regardless if you're a startup or an established company.

OUR PROCESS

Discovery

We work with clients to understand their business and we merge their initial concept with everything the Mobiversal team knows about mobile. We'll assess the possible challenges and identify the ways to overcome them.

Features & Architecture

We establish what features go into the product and how they will work together. Here, we're drafting a skeletal framework for the app in the form of wireframes.

Design

When it comes to first impressions, it's all about design. We'll put our passion for good design to work and based on approved wireframes we'll design all screens.

Development

The development process is broken down into sprints based on feature sets. Our agile-based development process will allow you to regularly review and assess what we're building.

Quality Assurance

Our Quality Assurance team will test the app after each development sprint and once all major functionality is implemented, we'll prepare a Beta Build.

Launch

After passing the Beta Build through a final round of QA and refinements, we'll have in our hands a Release Candidate Build. We can either submit the app to the App Stores or provide you with everything you need to do it yourself.

Maintenance

We provide our clients with a 3 months bug fixing period, free of charge, and maintenance packages that cover everything from small updates all the way to whole new versions.

Approach to meet requirements

The first thing we want to do is divide the whole project into major components. In this case these would be:

- **mobile app for catalog and taking orders**
- **restaurant web-based interface for active orders**
- **restaurant smartwatch app to be used by waiters**
- **administrator area with two levels authentication (employee and admin)**
- **backend (db and API)**
- **service presentation website**

More details about each major component are given below.

Design

1. Identity. App icon, logo & splash screen
2. Mobile app design for smartphones. 1242 x 2208 should work for smaller size screens with similar ratio
3. Restaurant web-based interface. Will be designed to run on touch screen PCs that have a printer connected.
4. Service presentation website

Mobile app

1. Detect if the user is inside or outside the restaurant. If it's inside the restaurant he will have small new features like partial payment or pinging the waiter.
    1.1. Search if the restaurant has beacon technology and choose to enter.
    1.2. Search using GPS to list nearby restaurants and choose to enter
    1.3. Request a table and wait for restaurant's approval in case this is the restaurant's setting. Restaurant admins have a setting that controls if users can access the restaurant directly or if they need employee approval to access the restaurant.
2. Menu
    2.1. List of menu categories. The menu will be grouped in categories, each category having a list of products or dishes.
    2.2. List of products from category. After selecting a menu category, users can see the dishes or products and can add them to orders.
3. Making an order while outside the restaurant
    3.1. Add to order from menu. Users can make an order even if they are not in the restaurant, but in order to receive it they should present themselves at the restaurant location.
    3.2. Current order. Everything the users select from the menu will be visible in the current order, before placing it.

3.3. Checkout order. During the checkout process the user can select the time when he wants the order ready.

4. Making an order inside the restaurant
   4.1. See all tables with their state. After accessing a restaurant the users can see the list of tables from that restaurant and can choose to occupy a free one or to request access for an occupied one. Each table from the list will have an info badge that the table will be occupied.
   4.2. Occupy free tables and choose how to let others join the table:
      4.2.1. setup a password and give the password to you friends
      4.2.2. receive a notification for letting the friend access the table
   4.3. Request access at an occupied table
   4.4. Add to order from menu. Having access to a table the user will be able to add new items from the menu to the current order.
   4.5. Current order. Users should be able to see the whole order of the table and his part from the order
   4.6. Selecting what products a customer wants to pay from the total. The users should be able to pay only for a part of the total items, this way they can pay only for their products.
   4.7. Pay with credit card from the app. Users can choose to pay for the order with their credit card. They can opt to save the credit card inside the app or to introduce the credit card details each time they pay.
   4.8. Like/dislike feedback for order. After paying for an order the user can choose to give back feedback
5. Register: name, email & password
6. Login: email & password
7. My profile
   7.1. User profile: email address, name, image
   7.2. My loyalty points
   7.3. List of my credit cards
   7.4. Add new credit card
8. Making reservation. Users should be able to fill a form in order to request a reservation. Once the reservation is approved they will receive notification.
9. My reservations. User will be able to see the list of all their reservations, pending and approved ones.
10. Other features
   10.1. Ping the waiter
   10.2. Change order if it wasn't processed
   10.3. Push notifications with promotions or messages sent by restaurant.

Website

1. Landing page. Will present the system and it's features
2. Register as restaurant. After registering, the restaurant will receive a shop

administrator account, described below in the Admin area section.
3. Login
4. Terms of service, Privacy policy

Restaurant web interface

1. Select current user. The restaurant can have multiple employees and when they access the interface they need to input their user code/password.
2. See list of tables. The restaurant employees will see the list of tables with their status and will be able to open the table details screen.
3. See table details
    3.1. Current order. The table details contains the current order with all its products
    3.2. Change table for order. It might happen that clients want to change their table for different reasons.
    3.3. Invoice and partial invoice. Employees should also be able to print the partial invoice.
    3.4. Make reservation. The employees will be able to make a reservation also.
4. See list of orders outside the app. Will be able to list all the reservations that are pending or active.
5. Create order. The employees should be able to create a new order that will be treated as if the order was done outside the app.
6. Menu. Employees will need the menu to create a new order.
7. See all reservations (accepted and the ones waiting for acceptance). It will list all the reservations that are in pending state or the ones that are next to happen.
8. Other features
    8.1. Info button that reservation will start in 20 minutes. Each table from the list will have a info badge that the table will be occupied.
    8.2. Approve reservation. A reservation from the clients have the time to start and the number of persons. The employee that accepts a reservation will assign a table to it.
    8.3. List of clients with their loyalty points
    8.4. Reset client loyalty points when giving him prize
    8.5. Ping the waiter. The employee that is using the restaurant web interface, might ping the waiter to go to the bar.
    8.6. Respond with approximate time to receive order.

Restaurant smartwatch

Employees would wear a smartwatch to receive notifications in real time and gain time, instead of having to check the restaurant's web interface. The notifications would be for:

- when a user pings the waiter
- when the bartender pings the waiter
- when an access restaurant request is received

Admin area

Three level authentication:

1. **Employee**
   1.1. Manage products. The employees should be able to login the admin area and make the stocks. This means they should list, add, remove products.
2. **Shop administrator**
   2.1. Manage employees. The shop administrator will be adding, removing the employee accounts.
   2.2. Manage tables. The shop administrator will be adding and removing the tables.
   2.3. Analytics. The shop admin will see the list of all the orders made in his restaurant, grouped per month.
   2.4. The shop admin will be able to manage products as well.
   2.5. Edit restaurant settings. The name, image, enable/disable employee approval for entering restaurant, Money spent to Loyalty points relationship,
   2.6. List of feedbacks
   2.7. Send a message to all the customers
   2.8. Configure payments settings to enable mobile payments
   2.9. List of shop invoices
   2.10. Pay for subscriptions
   2.11. Cancel shop account
   2.12. Contact support
3. **App admin**
   3.1. List of all shops
   3.2. List of all invoices
   3.3. List or all support request

Backend

1. Architecture and structure
2. Server setup. We would prefer to use LAMP model for this type of project.
3. API
   3.1. User
   3.2. Restaurant
   3.3. Push notifications

Technical solution

We are using RUP (Rational Unified Process), so the development of parts from the components can be done in parallel. The diagram from the costs and timeline will explain how the components are done in parallel.

Technologies proposed for the development:

1. **Backend and Database**

We have selected a Javascript-based stack to implement the backend, based on NodeJS, ExpressJS and SailsJS. As opposed to more traditional technologies, Javascript is a newer one but growing quickly on the server-side, and we selected it because of its stellar performance and better scalability.

The database storing restaurants and products data will be implemented as a NoSQL database, as that data grows quickly and a traditional SQL database will not provide support for the needed scalability. The database system will be MongoDB as it has extended support for location processing (spatial indexing), and additionally it integrates smoothly with NodeJS.

2. **Admin Area and Website (web client)**

We are using AngularJS 2.0, a modern Javascript-based framework to implement the website.

The admin area will have a simpler UI and functionality and will be implemented using a template based on Bootstrap 3.0.

Both the website and the admin area will run on all major current browsers: Chrome, Firefox, Internet Explorer.

3. **Mobile app**

The Android mobile client application will run on all phones with Android 4.0 or newer that include Google services. The iOS mobile client application will run on all devices with iOS 8.0 or newer.
It will be developed using the native development tools provided by Apple and Google and will use Google Maps for some features (nearby locations). The app will run in portrait mode and will have a common layout and look-and-feel across all form factors (phones and tablets).

During the development phase we propose at least the followings:

- Each week we are delivering demo files (images, web-pages or mobile demo builds)
- Each week we must have a general sync meeting to discuss:
  - last week's progress based on the demo provided
  - next week plans
  - other items regarding that period

Tools used by Mobiversal:

- JIRA for ticketing and project management
- Worklog Assistant for detailed time tracking
- Email and Google drive for sharing document, specifications and content
- Crashlytics Fabric for crash reports on mobile
- Crashlytics Beta for beta testing environment

Timeline and costs

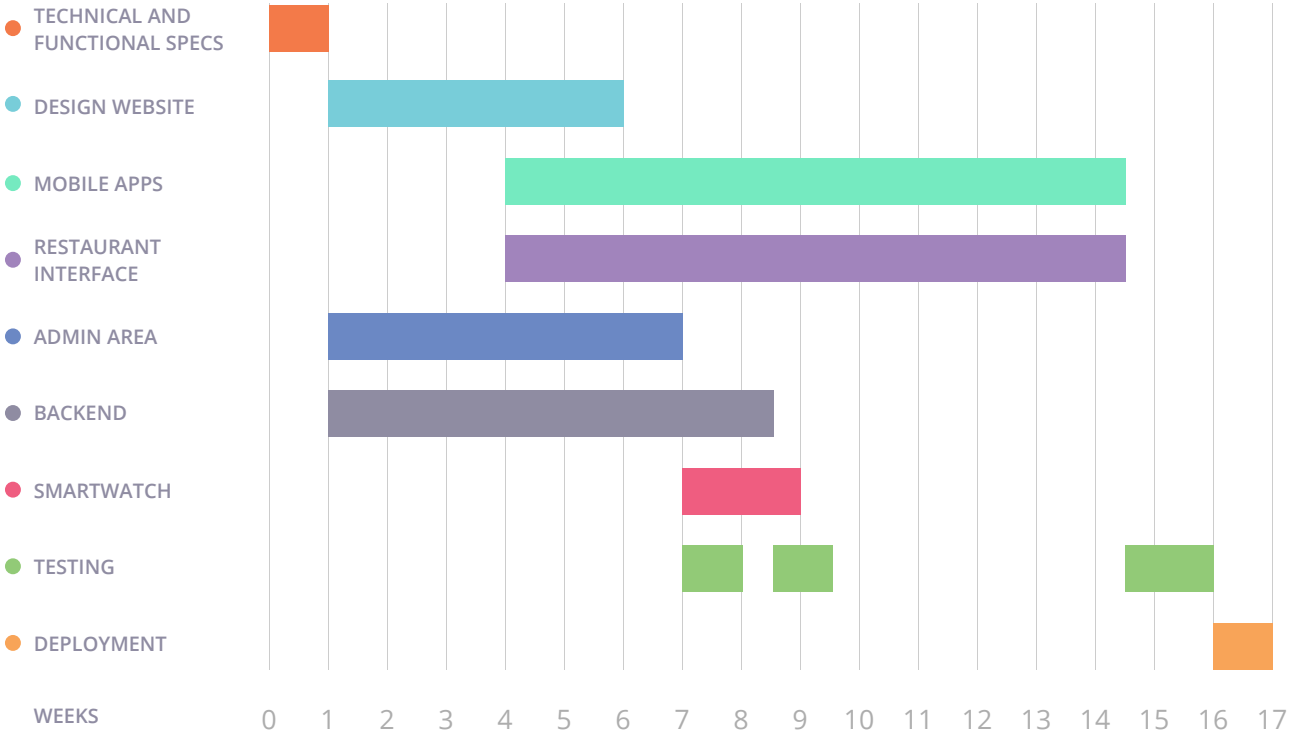
Before starting the actual designing and coding, we will need to build a full functional requirements documents. This document will contain everything the system will be able to do with higher level of details, because it will be the way to check and compare everything needed to be done. The designer, software architect, software developer, QA team, product owner, and project manager will know exactly what to build and what to expect from the system. This part should not take more than 1 week and will require communication between our technical business analyst and the product owner.After this scoping week, the designer and the software architect can start working. After having the architecture, the developers will also be able to start developing.

During the development we will have two important milestones: alpha milestone and beta milestone. For the alpha milestone, around 40-60% of the features should be working and for the beta milestone 100% of the features should be working, but with chances of bugs.

After the beta milestone, the apps will be sent for QA testing and fixing found bugs. After fixing this bugs, the apps will be ready for your user acceptance testing.

After the user acceptance testing is ready (which means you've tested and found that everything is ok), it starts a warranty period of 3 months. The total estimated time of development is between 16 and 17 weeks. Below you can find a timeline chart for the whole project, with each component apart.

Timeline



Costs

TASK	DEVELOPMENT PERIOD
Design work	6 weeks
Backend	7 - 9 weeks
Admin area	8 weeks
Mobile app iOS	8 - 13 weeks
Mobile app Android	8 - 13 weeks
Website	1 week
Web-based interface	6 weeks
Smartwatch app	2 weeks
Project management	100 - 120 hours
QA Testing	3 - 4 weeks
TOTAL	16 - 17 weeks
	TOTAL: $65.000

Do you want to have this project done by Mobiversal or do you have a new challenge for us?

We love taking ideas and turning them into real apps. Tell us your idea and we'll give you details about costs. Simply click below to tell us more about your project.

Get a free quote