BUILDING PHOTOKAST

Creating an iPhone App in One Month

Introduction

Ten23 Software is a company that specializes in web/mobile development and consultation. We're a small company consisting of two principals and little free time. We have worked on many different software solutions, from GPS-based truck tracking systems to mobile business intelligence tools to social networking sites. **A few months ago, we challenged ourselves to create a social networking iPhone application in 30 days and PhotoKast was the result.**

I remember when we got our start in mobile development - answers were seemingly inaccessible and everyone who claimed to have them *usually didn't*. Sometimes, all you need is a little head start. This document provides some tips that we've picked up in the last four years, with examples drawn from the recent development and launch of PhotoKast. Much of the content speaks to mobile development in general, with some specifics regarding the iPhone platform. Since you can find a lot of coding material on the web, I'll focus more on the other questions that get less coverage. There is a lot to know about the design, development and promotion of mobile apps…and you certainly won't find it all here. Also, there are varying opinions on how best to do it. For these reasons, I've included links to other resources at the end of this document that will hopefully point you in the right direction.

This document will illustrate some things we have learned through "15 tips". We're not claiming these to be a magic formula that will guarantee you success (that will stem from your idea and how well it's executed) – they're simply suggestions pruned from experience. This is our contribution to those getting their feet wet in mobile development; we hope you find it helpful.

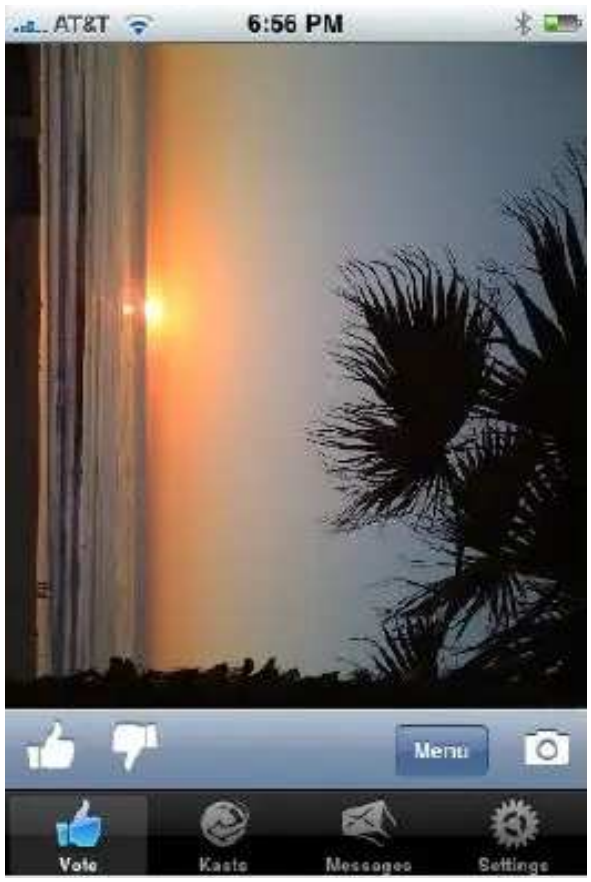
The Concept

Our concept for PhotoKast was simple: **allow users an easy way to share their photos and track their worldwide popularity.** When a user ("kaster") broadcasts a photo, it is sent to other PhotoKast users in the area. If they like the photo, they can give it a "Thumbs Up", which will then send it on to other people near them. Hence, the more people that like the photo, the faster and further it will travel. For each photo, users can view a "Photo Trail" map which shows every place throughout the world that people enjoyed the picture.

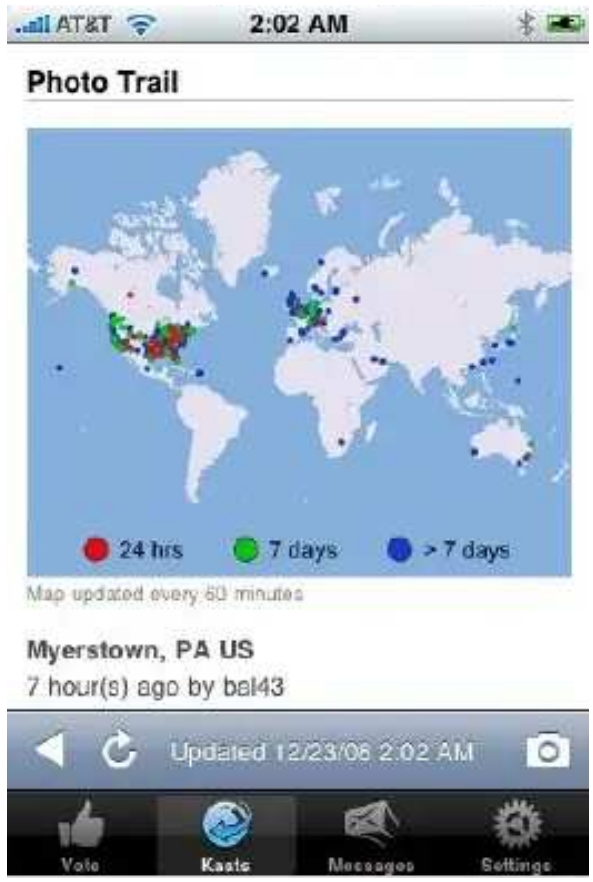
The final requirement? It had to be done in a month. That includes design, database work, server code, spatial implementation and of course, the iPhone application itself. Any features that couldn't be incorporated within the month time limit would have to wait…perhaps indefinitely.

The Result

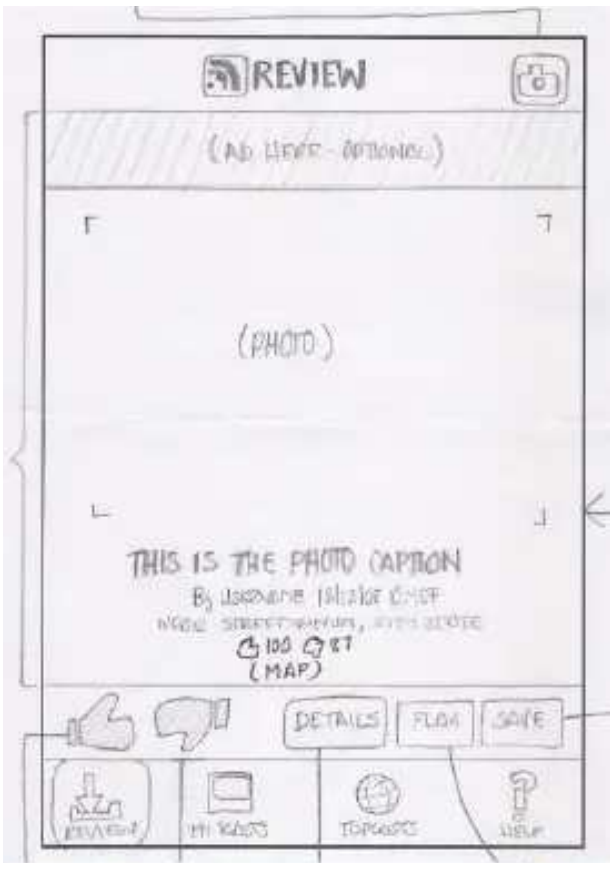
PhotoKast turned out to be much more popular than we anticipated. Just through word of mouth, we quickly registered tens of thousands of users and are serving well over 2 million photo views a week. Within the first month, the paid version of PhotoKast was one of the top ten social networking applications. It's certainly not the runaway hits some iPhone apps have become, but still not bad for a couple of programmers and a month of development time.



Users can look through photos in their favorite categories. They must vote to see the next one.



A photo trail accompanies each photo, so you can read comments from other users, and see where it has been most popular around the world.



Vote Tab initial design sketch

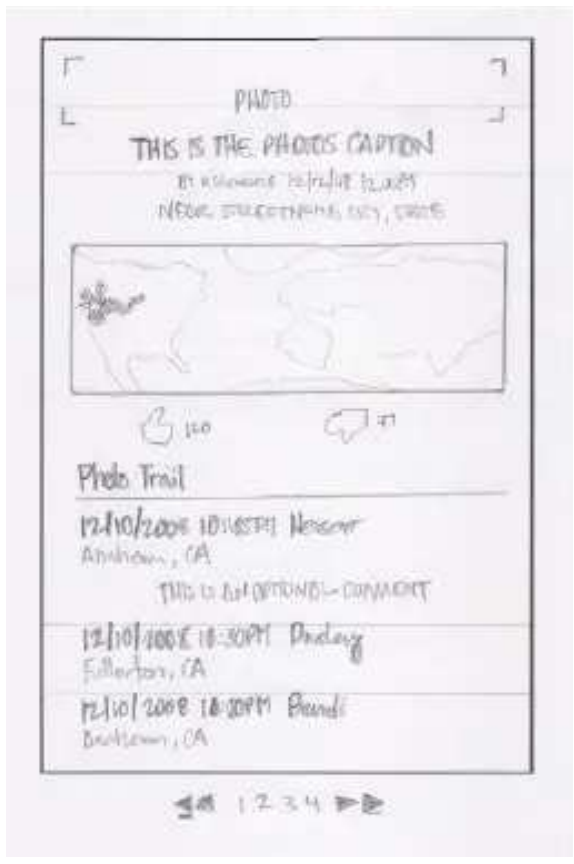


Photo trail initial design sketch

15 Tips

iPhone. Start here.
The iPhone is a great platform to build a mobile community

What's in a name?
Discovering a good name for your product is important

Designing the 7 deadly sins
Make sure your application concept is addictive

Easy as 123. Or not.
Make sure your application is intuitive and easy to use

Short term commitment or marriage?
Think about the shelf-life of your product *before* releasing it

Be Realistic, Part I
Consider the possibility that you won't hit 1 million users the first month

Be Realistic, Part II
Be realistic in how you intend to monetize your application

Make believe you will succeed
Define a path for future scalability

UGC = T&A. A law of nature.
If you display user generated content, expect body parts

Be embarrassed by your first release.
Don't try to make your product perfect before releasing it

…But not too embarrassed.

Do make sure it has the essentials

It’s hard to draw a line in the sands of a slippery slope.

Define your standard for acceptable content, and stick by it

Sometimes, they write the owner’s manual.

Sometimes they use your applications in ways you never intended

Footwear, software and customer care.

Practice good customer service

Room for improvement (there always is)

Mobile development is in its infancy

iPhone. Start here.

I'll admit it. When Apple released the first iPhone, I was a skeptic. I blogged about the "unreasonable" expectations of 10,000,000 phone sales in the first year, primarily because it was priced so high and lacked support for 3rd party applications. Even with the price drop that occurred shortly after, sales fell short of expectations.

But I also talked about how sexy it was. "*Jessica Alba* sexy", if I remember my words correctly. It was revolutionary – not necessarily because of the technology – but because of the way it wrapped that technology up in one smooth, easy to use package. It wasn't the first touchscreen, it wasn't the first to sport an accelerometer, and it didn't have the best camera on the market. But it looked great, and worked even better. And that had more to do with the software than the hardware. Its usability rocked. And competitors who strove to beat it with more advanced hardware still failed on the key point – software. Don't underestimate the OS.

Then the clouds parted and rays of sunshine beamed down upon the multitude of mobile developers once shrouded in isolation and darkness. Sound dramatic? *Then you probably aren't a mobile developer*. Apple released an SDK so 3rd party developers could write applications for their phone. Now, don't get me wrong… there already were other platforms that encouraged 3rd party development. But they didn't have the buzz of the iPhone, nor the application distribution reach. The App Store isn't perfect, but it sometimes seems so when compared with some of the other options. Apple kicked the industry in the pants, and these new technologies such as Android point towards a bright future.

> We recently had a customer support request asking us how to move the PhotoKast icon to a different location. If you think about it, that's pretty amazing. For a long time, most consumers didn't understand that their devices could be used for much more than voice and text messaging.
>
> Now, even users unfamiliar with some of the basic operations of their phone were able to browse, download and launch 3rd party applications.

Bottom line? Choose the platform that best serves your target audience. If you are building enterprise applications, you will likely be looking at the BlackBerry (http://www.blackberry.com/developers). If you are looking to build an application that has the furthest worldwide reach, you need to check out Nokia (http://forumnokia.com). If you're looking for an audience of progressive users that are heavy data consumers, definitely consider the iPhone (http://developer.apple.com).

What's in a name?

We chose the name "PhotoKast" for three simple reasons:

1. The domain was available.
2. It was self-explanatory.
3. It could be "verb'd" ("I kasted a photo", etc.)

Obviously, name your product well. Poorly named products can still succeed, but a little time dedicated to discovering a good title for your product may help you tremendously. Domain names are hard to come by, but that's no excuse for calling your app "MyyFreindsz".

As we'll get to later, determining a "hook" for your application is an important part of the design process. What makes your application addictive? Why will people want to use it? The hook starts with your application's name.

Applications such as "WhosHere" and "PhotoSwap" clearly benefit from the self-descriptive nature of their well chosen names.

Designing the 7 deadly sins

"Build your social network for the seven deadly sins."
Tim Chang, Norwest Venture Partners

Tim Chang, from NVP, spoke last year at a little Web 2.0 event held in Orange County CA. There, he perfectly summarized a key design rule for social networks: build them for one or more of the seven deadly sins. Thanks to Wikipedia:

Lust	Obsessive or excessive thoughts, usually sexual in nature
Gluttony	Over-indulgence, over-consumption
Greed	A sin of excess like lust and gluttony, but in reference to wealth
Sloth	Laziness, indifference, apathy
Wrath	Uncontrolled feelings of hatred and anger
Envy	Resenting another because they possess something you do not
Pride	Excessive love of self

I'm not saying you need to write an application that encourages one or more of the above, and I don't think Tim was either. Personally, I'd discourage it. But think about ways you can use them constructively in applications:

Lust	Dating
Gluttony	Shopping
Greed	Budget, Financial
Sloth	Productivity
Wrath	Debate, Politics
Envy	Shopping
Pride	Online Identity Management

So, for example, you can create a personal productivity application that saves people time and money. This will appeal to facets of the "7 deadly sins", but not in such a way that encourages and promotes behavior you don't personally endorse.

Hook'em

Before you even come close to coding, you need the concept for your app to be flushed out, and it must include a hook. What is it about your application that will be addictive? Why will people pick it up and not put it down? If you look at the reviews for PhotoKast, you'll notice that even the negative ones

typically compliment the concept behind PhotoKast (the negative reviews often had more to do with other users). If you look at the positive reviews, you will see one common theme: *PhotoKast is addictive*.

What makes it addictive? It appeals to our nature. We are a curious animal, we like to see what others are up to. With PhotoKast, you view one picture at a time, and you always want to see the next one. You may have to march through a bad batch, but you'll eventually encounter one that you like…and then the process repeats. When we discussed the idea with friends, some recommended that we include a lot of animations and UI polish to keep users engaged. We knew that if we had to do that to keep a user's interest, our concept wasn't strong enough.

We don't claim to have constructed an elaborate, addictive design. In fact, it's quite the opposite. PhotoKast is based on such a simple premise it's ridiculous. But that's also the point. I remember back when games started to approached Hollywood proportions; long 3D animations, full soundtracks, etc. Like a lot of the movies, production value went way up…but game play suffered. Then, along comes Tetris – a simple game built on a simple premise that lacked the musical scores and interstitial 3D scenes of the bigger budget competitors. But it was fun. Occasionally, we'll get clients that want to create a mobile app that has every feature under the sun, eg. a single app with 14 cheesy games built into it. This approach already signals defeat; it says that you don't believe you have a strong concept, and so you try to throw everything in hoping something will catch. Don't do it, you'll be wasting your time. Instead, go back to the drawing board and distill everything down to one core concept that is inherently addictive and go from there.

Make Celebrities

Mark Twain once remarked that there is no such thing as a boring life. Everyone has a story dying to be told. We all want to feel important, appreciated, understood and supported. This is a primary attraction of social network sites: validation. Validation of our thoughts and opinions and stories and knowledge and capabilities and looks and popularity. As a designer of a social networking application, your job is to create celebrities.

PhotoKast doesn't pretend that it will get you a Hollywood agent or your own sitcom. It does allow you to easily take a picture of your favorite place, event, person (including you) and see how popular it becomes throughout the world. Within this little ecosystem, there are already several well known users: Robertandrewbarnes, killer, Lantern Rider, thatGIRL, Dancebarbiexoxo, SiliAlley, Bernyboymk, u.heart.moi, cchw1 and more. They've kasted popular photos and have earned their top spots. What you will want to be mindful of is how you reward users for being active in your community; how will you bring positive attention to them and make them celebrities of that little world?

Creating celebrity is not just about putting the spotlight on your contributing users, but also giving them the tools for reputation management. Admittedly, this is not something that PhotoKast did well when it was released. PhotoKast users can add comments to other people's kasts. Unfortunately, the author of the photos could not remove negative comments left by others. This would sometime lead to the deletion of a kast because of one brutal comment. Because we gave

ourselves a one month time limit, we intentionally left this off the list of features that would make the first release. We would make the same decision again, but that doesn't mean you should; consider the importance of reputation management to your application and implement accordingly.

Follow the Formula

Have you seen the movie about the guy who suddenly changes his personality, and it causes him to realize all the good things he missed before? This might sound like Jim Carey's character in Liar Liar. Or it might sound like Jim Carey's character in Yes Man.

Or what about the tale of a family's Christmas get-together that includes a wife (or girlfriend) that doesn't fit in and the eventual confession from a parent suffering with cancer? I could be talking about the Family Stone or Home for the Holidays.

Or maybe you've seen an episode of CSI: Miami – *any* episode of CSI: Miami – and are quite familiar with the start of each show…a survey of a crime scene that also shows Caruso, decked out in his familiar shades, ready to deliver the one-liner that ushers in the series' "heeeeeeey" theme song.

Hollywood knows how to repackage formulas that work, and you should too. Don't reinvent the wheel; sites like Facebook, MySpace and Flickr have already dialed in on features that work. Remember Facebook's flop with Beacon? That was a lesson learned at their expense. Find out what they do right and figure out a way to implement those popular features in your unique design. We're not promoting lazy copying, but as popular quotes such as Newton's "…on the shoulders of giants" or Picasso's "bad artists copy, great artists steal" imply, there is abundant precedent for you to learn from.

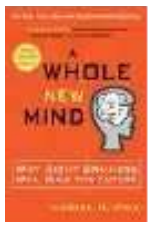
Women Rule

No doubt about it, women rule. In the online world, marketers court them and social networks need them. Think about sites like match.com. Any female on there with a decent picture is bound to get a ton of winks from their "eAdmirers". An attractive guy, though he may be popular in comparison with his peers, will likely get much less activity.

It's easy to attract guys to your product; have a lot of female members. Attracting women to your product is a much more complicated endeavor, but one you'll want to concentrate on. Are you creating a social network dedicated to women's interest, or is your application for both men and women? To make your site a comfortable place for women to participate, you'll need to consider reputation management and functionality that speaks to their needs. Don't let them be bombarded with messages from would-be courters and let them dictate the pace of interaction.

Easy as 123. Or not.

You would think that applications that are easy to use must be easy to develop. Contrary to popular opinion, designing for simplicity is anything but. Users of enterprise software know exactly what I'm talking about; the interface typically models the underlying database structure more than their actual workflow. An application is not great because developers demonstrated unsurpassed dedication to the principals of object oriented development. It is the design that makes it great. Coding is a commodity, creativity is a premium (well, *good* coding is still a rarity). Software companies would do well to afford their developers the time and tools to cultivate their creativity.



An interesting read:
Dan Pink's "A Whole New Mind"
http://www.danpink.com/wnm.html

I've often told clients who are looking to build a web site to first think about how they would design it for a mobile device like the iPhone…even if they have no intention of going mobile. Since phones have limited resources, user input capabilities and screen real estate, designers must make hard choices about what is essential to the application and what is fluff. This simplified set of functionality is an ideal foundation for the web site they intend to create.

Simplicity was a key design objective for PhotoKast. Basically, a user is presented with a photo and must give it a "thumbs up" or a "thumbs down" to see the next one. If they give it a "thumbs up", then they can also leave a comment for the photo. That's it.

Some users have asked us to allow them to also be able to comment when they give a photo a "thumbs down". It was a design decision early on to only allow commenting on up votes. Why? When a user gives a "thumbs up" to a photo, it increases its overall vote score AND helps it to travel further. At this point, a user that wants to leave a comment like "your photo suuuuucks" must actually help it travel further in order to do so. It's a nice little dilemma that helps curb (but not totally eliminate) negative remarks.

Short term commitment or marriage?

Before you release your application, make sure you consider how long you plan to be wed to it. Some applications have a defined shelf life; for example, the Obama iPhone application* was intended to help the President's campaign by keeping his supporters connected to the resources they need. And of course, a way to easily donate. But most applications do not have a defined end point, and you need to consider the long-term plans for keeping it afloat.

Do you store media? Do you need moderators? What will you do about customer service? How will you accommodate more users if you're fortunate enough to have that problem? How will you generate enough revenue to pay operating expenses, and a little more to make it worth your while? How will you keep it afloat when the novelty wears off and you generate less revenue but operating costs remain the same (or increase)?

* I never downloaded the application, but had seen some of the screenshots and it looked beautifully done. I believe it was created by a team of 12 people in under a month.

Be realistic.

Be realistic about the traction you think you'll get after your release. It's good to be enthusiastic and optimistic about your product, but realize that it's not easy attracting and keeping an audience. Think about it this way:

Success = Execution + Opportunity

Execution: Hard work, skill, design. Things you can control.
Opportunity: Timing, connections, risk, luck. Some you control, some you don't.

You could be one of the fortunate ones – you hook into a great idea at just the right time and traction comes easy. Or, you could be like most, where you have to fight for every little bit. You can control the execution aspect of your project, ensuring that your product is complete, bug free and the bee's knees. But there is a certain amount of luck and timing outside of your control that will help dictate your path. In our experience, most don't even move beyond the execution part of the equation. If you do, be willing to work for it should you not get the windfall of registrations referred to in your investor pitch.

As I write this, there is a documentary on a young country singer named Taylor Swift playing in the background. I'm not such a big country music fan, but I'm a fan of hers if this documentary is true. She had a vision and worked hard to achieve it. Apparently, at 14, she walked away from the biggest country music label because they wanted to sign her to one more year of development, rather than give her a shot at her first album. In retrospect, it was a great decision. But it could just as easily been a terrible move. At 14, she took the risk and it paid off. Entrepreneurs face those types of decisions all the time. Experience helps, but what successful businessman has not also profited from ~~a little~~ luck and opportunity here and there?

Be realistic. Part II

At some point, you're going to need to make money. The typical response is to generate ad revenue, and the second most popular response is a subscription fee for extended functionality ("freemium"). Be realistic and don't assume mobs of people will immediately recognize your product for the divine gift that it is, with advertisers trampling each other on their mad dash to your office door. Here are some points to keep in mind when brainstorming your business model:

- **Determine your path to monetization.**
  Will you display ads? Will you give it away for free but charge for extended functionality? Will it be a subscription service? Will there be a one-time cost for the application? Many apps have benefited from offering a free version along with a paid…just don't give too much away in the free.

- **Build in hooks for monetization early.**
  However you choose to monetize your application, build in the hooks early. If you will be displaying ads, make sure the code is in place to display those ads before your first release. Even if you don't display them initially, you'll appreciate the latitude to be able to toggle them on whenever you want without having to push out a software update.

- **You probably won't get rich off of the App Store revenue.**
  PhotoKast was in the top ten paid applications for social networking in the first month, but it still hasn't really made us much money. There are a few applications that have sold huge, but they are few and far between. Games have wide appeal, and so if you build an appealing one and happen to land in the "Featured" apps list, you're life – at least for now – is pretty good. If you're building a social network, remember that you will probably get fewer eyeballs and are competing with (much) larger, well established entities.

- **People expect to give little and get a lot.**
  Users have been cultivated to expect things for free. MySpace and Facebook are free. The App Store is inundated with many free and cheap applications (see http://furbo.org/2008/12/09/ring-tone-apps/). Your challenge is to build a product that people are willing to pay for, or use enough so that an advertiser will pay you. The challenge is likely even greater because there probably is another application just like yours being offered for free. It can be crap, but users will point to it as a model that you should follow. This is unfortunate; budget apps condition users to expect everything for cheap/free, and this climate dissuades developers from spending a lot of time on apps because they can't make their money back. Hopefully, Apple will eventually highlight "Premium" apps to help curb this trend.

- **You'll have to serve A LOT of ads.**
  We had ads running in PhotoKast for a very short period of time. There was a bug in the ad code provided by the ad network that created a poor user experience for our customers so we quickly backed it out. But in that period of time (less than 24 hours), we made 69,336 ad requests and generated a whopping…wait for it…$16.04. I guess the beachfront property in Hawaii will have to wait.

- **Ads are a philosophical battle**
  If you are going to run ads, run them from the start – or at least prep the users from the start. The last thing you'll want is to let your customers get used to your product without ads. Then, once you reveal them, you're sure to get a few irate users who completely object to the "corporatization" of your product. 95% of your audience will understand your need to generate revenue through advertising if the product is free (as long as they are not intrusive), but there will be a small percentage who take offense and could care less about the economic realities. It's not about the 320x44 pixel space they give up. It is the philosophical view that ads represents some sort of capitalist greed that runs contrary to the spirit of sharing, altruism, open source, etc. Avoid this conflict by building them in from the beginning.

- **Consider operating expenses**
  Be sure to factor in your operating expenses. If you charge a one-time only fee of $3 per application, make sure that it can cover storage costs, hosting fees, etc. over the lifetime use of the product.

- **Consider other operating commitments**
  Will you need to employ moderators? Are they working on a volunteer basis or will you compensate them? How much time will you personally have to dedicate to the application? Is your application fairly self-maintaining?

Make believe you'll succeed.

What happens in the event that your mobile application is a resounding success? Besides cashing big checks and planning bigger vacations, you'll need to make sure the application itself stays tuned up and running like a champ. You'll need a scalability path defined so you don't burn the midnight oil trying to figure out how to accommodate all they people that like your product. Here are some things to keep in mind:

- **Hosting.**
  If your application requires a server, then you will need to think about where you will host it. A clunker PC in your bedroom or a tired old Compaq in the back of the office is probably not the ideal start. If you choose to use an inexpensive hosting plan, define how you will scale it up and out to accommodate more people. Don't rush out and overpay for service and bandwidth that you don't need; do understand how you will keep pace with the volume as your audience expands.

- **Storage.**
  Will you store media , such as video, photos, etc.? Make sure this component of your service can scale too. Popular services such as Amazon S3 (and their new content delivery service) are popular with a lot of startups who need cheap space. It's a well tested service and very affordable. Just realize that you don't have complete control over uptime.

- **Database.**
  Even if you store your media in a storage service like S3 or Nirvanix, you will likely still need a database to keep track of user accounts, comments, etc. With PhotoKast, we keep track of every vote from every user. The process is somewhat complicated: when a user votes, we must record that vote in the database with their comments and the location of where they voted (we use their location to build the photo trail maps). Then, it needs to look through all of the kasts for the ones that have been getting favorable responses and choose the best one (by looking through millions of votes) that has not yet been voted on by the user in question. This whole process occurs in about 10-30 milliseconds. Pick a database that works for you and then be prepared to analyze and optimize it as it grows.

- **Backup / Restore.**
  If you're storing data in a database, or are storing your own media, then you will want to have a good disaster recovery plan in place. I live in Southern California, and every year when the Santa Ana winds pick up and the air is dry, inevitably fires begin to consume the brittle hillsides and subtly encourage us to pack up our most prized possessions. The day after, I'm on the phone with the insurance companies making sure that my coverage would've been sufficient. That, my friends, is **not** the way to conduct your disaster recovery plan.

UGC = T&A. A law of nature.

[Note: since we will be talking about user generated content, we must also discuss the type of offensive content you will almost certainly see should you create a UGC application of your own. If you find the topic too offensive, I would suggest that you skip this chapter and also strongly reconsider creating a UGC site/app. Even if you are putting together a social network for ferret lovers, someone will upload something that you won't want to look at.]

PhotoKast was released on a Sunday night. We saw a few users trickle in from overseas (at that point, I still didn't see it listed in the App Store on my phone). It was late, so I put the phone down and went to bed. I woke up at 3 AM and decided to check out the app before going back to sleep. I turned on the phone, started PhotoKast, and saw a new photo from a new user.

Of that new user's breasts.

And it wasn't the last of its kind. Needless to say, it was difficult returning to sleep. Based on our experiences with other social networking projects - and common sense - we anticipated that some users would attempt to post "offensive" content from time to time. We implemented a flagging mechanism whereby any member of the community could tag an image for review. Flagged photos are immediately taken offline until a PhotoKast admin reinstates it or cans it for good.

But of all the UGC sites we've worked on, PhotoKast easily has the highest frequency of offensive content. This surprised us a bit, but in hindsight makes perfect sense. While many social networking sites are built around a user's identity, PhotoKast is much more anonymous. Hence, people feel a little more free to let it all hang out. We're pretty sure people looking for great pictures of Hawaiian beaches or Renaissance statues in Florence probably didn't want to see a 320x480 pixel view of someone's genitals. Just a guess. For the week after the release, we developed and implemented a fairly sophisticated system for community moderation. Since that time, the moderators (volunteers from the PhotoKast community) have done an excellent job. 20-25% of photos kasted violate the content posting policy and are prevented from being publically viewable.

Why didn't the flagging system alone suffice? Isn't it enough to give the community the tools to police itself and let it be?

Under normal conditions, a simple flagging system may have been sufficient. Not with PhotoKast. The popularity of the app and its broadcast-nature meant that a large group of users would see the photo the instant it was posted – too many people for our taste. If it didn't get flagged by that group, it would travel on to even more people until someone would eventually flag it.

Content Moderation

Since PhotoKast is based on the concept of broadcasted images (rather than people seeking them out), we decided a simple flagging mechanism would not be sufficient. Within the first week after release, we started implementing a more complex form of content moderation. PhotoKast users were then divided into three groups:

- **Administrators.**
  Administrators have the ability to flag photos, override any moderator decisions and suspend users. They can perform these duties from the PhotoKast application on their phone or from a special web administration console.

- **Moderators.**
  Moderators have the ability to view pictures before they are released to the public, and determine if they should be released. They can perform these duties from PhotoKast on their phone. They do not work for us or represent us, but have volunteered to perform PhotoKast community service.

- **Standard users.**

  Standard users do not have any admin or moderator capabilities.

We selected moderators from the PhotoKast user community and gave them the ability to preview photos and determine if they should be released to the public. Upon moderator approval, the photos are immediately broadcasted to PhotoKast users.

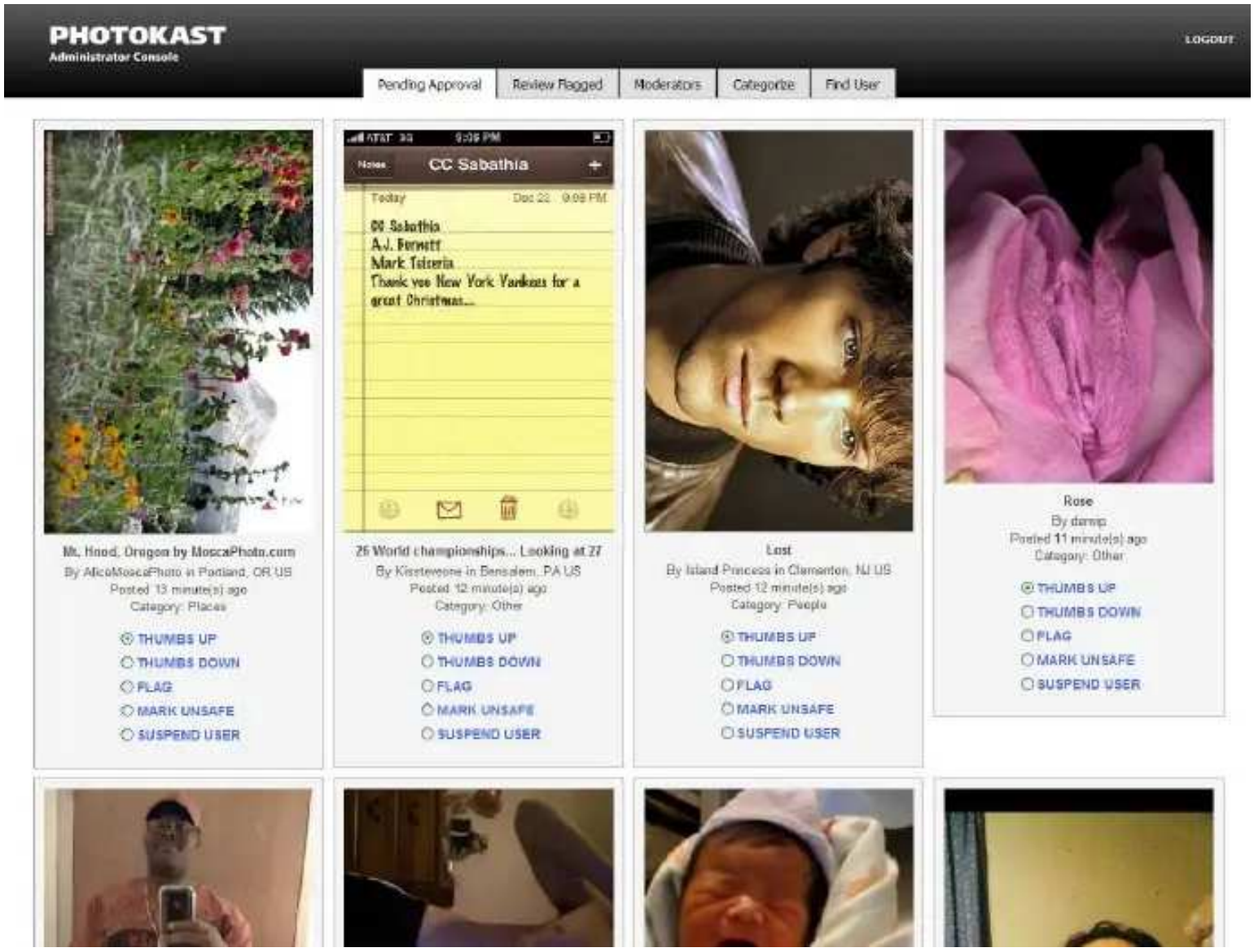
The Process

If any member of the PhotoKast community flags a photo as offensive, it is sent back to an administrator along with the name of the moderator who approved it. If the photo should have not been released, then we warn the moderator. If it is the second offense, the moderator usually loses his or her privileges. The administrator then makes the decision of whether the photo is truly offensive or not. Once an admin has decided, no amount of flags can overturn the decision.

That being said, the moderators have been doing an outstanding job. Usually, when a photo is flagged by a user, it is because they don't like it. They don't agree with it politically, don't like the person who kasted it, thinks it's ugly, etc. But flagging a non-offending photo actually has the opposite effect of what the user intended. Even though the photo is temporarily taken offline, once an admin marks it "safe" it immediately gets priority broadcasting status and reaches more people than it otherwise would have. By flagging a safe photo, the user has inadvertently given it a bigger audience.

As you can see, there is much more to the system than users realize. A brief tour of the Administrative Web Console (that only admins have access to) and the phone-based content moderation tools are provided over the next couple of pages.

Administrative Web Console



Pending Approval Tab. This tab shows all of the photos that have yet to be released to the public. Typically, moderators approve/disapprove of these images from the phone, but admins can also do it from this page of the admin console.

Review Flagged Tab. This tab shows all of the photos that have been flagged by the community, when they were flagged, who flagged them and which moderator approved it. From this tab, admins make the final decision for the photo (re-broadcast it with priority or keep it offline).

Moderators Tab. From this tab, admins can see all of the photos each moderator has viewed, and their decisions for each. This is the tab the administrators use most often.

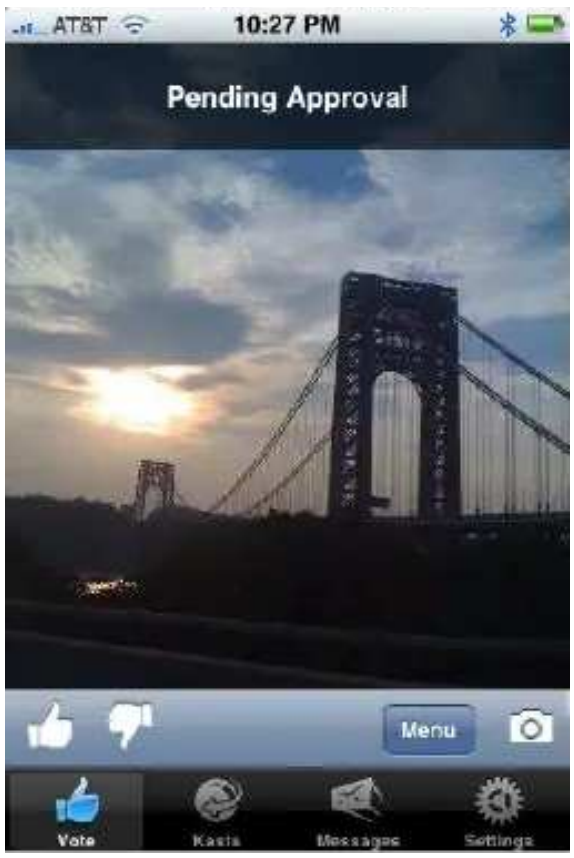
Categorize Tab. From this tab, admins can categorize and re-categorize photos that may have been posted in the wrong sections.

Find User Tab. This tab lets admins look up user information.

Admin and Moderator Phone Tools

Standard users get a button to private message and another to view the worldwide popularity map.

Admins get an extra two buttons to suspend a user And to mark a kast unsafe.



Moderators can see photos before they have been released to the public (“Pending Approval”)

Revenge Reviews

If you decide to start a social networking application, be prepared for two things:

1. You will find that your app is reviewed based on the community as much as it is reviewed on its own merits. Some of your users may love the concept of your app and how it works, but become disenchanted with the other people who are using it and grade you according to the latter.

2. Revenge reviews. Since you are building a community of users, you will no doubt have to suspend or ban those who choose to violate your terms and conditions. Expect that a user ban will almost always result in that person posting a negative comment about your app. Don't let that dissuade you from doing what you have to do.

As of the time of writing, PhotoKast and PhotoKast PRO both have a "4 out of 5 stars" rating in the App Store; it is one of the highest rated social networking apps in the category. We're content with this, given the difficult nature of building a solid app and keeping a generally happy user-base. Still, we don't mind risking the potential of revenge reviews in order to get rid of those who choose to negatively impact the community experience.

Comment #36: by "Unexceptable"
This is a revenge review from a user who was caught posting explicit content and stole other people's pictures and passed them off as his own. When he was confronted, he threatened to leave a bad review about the application if we removed him. His review essentially criticizes the conduct he was banned for. Unfortunately, the casual reader does not know the autobiographical nature of his assessment. Still, removing him was worth the retribution and it's a better community without him.

Review #28: A great review, but they accidently gave it only 1 star. Don't you hate when that happens?

This user's photos stopped receiving votes because other user's stopped voting for them. If a photo recieves votes, it means it made it passed to moderators.

Be embarrassed by your first release

"If you're not embarrassed by your first release, then you launched too late."

Reid Hoffman, LinkedIn

A lot of bad software exists because developers had low standards for release. Obvious bugs and terrible UI have plagued users since software has been around. But there is also the opposite problem…*expecting too much.* What a lot of people don't understand is that software development is an art form. There must be vision. There must be design. There must be a good user interface, which is especially artistic if it is a game. Possibly a soundtrack. A good software application is the marriage between creativity and hard work. And so for many developers, they want it to be just perfect. A masterpiece at version 1.00. This is a mistake.

The customer will help you design

If you're a small company, then waiting too long to release can be fatal. Use your agility to release early and adapt to feedback. Let the big companies worry about extended release cycles, focus groups, etc. Put your app together, make sure it contains the essentials, and set it free as soon as possible.

When you design your app, make a conscious decision to determine what is the core functionality and what is a nicety. What features must be included for the concept of your app to be properly conveyed? Everybody begins with a huge list of features that would make it a great application, but it's not important to get them all in initially. That's why we have versions.

Abstaining from implementing your entire feature list may prove beneficial in the long run. With PhotoKast, we had a list of features we thought users would really enjoy and we had a list of quirks we thought would irritate them. Turns out, after release, our users didn't mention some of the features we thought they would request, they requested some features we didn't think of, and barely complained about quirks we thought would stir more frustration.

The lesson? Your customers understand their wants and needs more than you do. Release your application as soon as possible, be attentive to customer feedback, integrate good suggestions quickly, release an update…and repeat. This cycle ensures that the product is conformed to the audience it was written for. For customers, there are few things better than to offer suggestions and then to see their input materialize in the next release. They appreciate being able to contribute to the evolution of the product and as a result, often become evangelists for your application.

6. …But not too embarrassed.

You will need to decide which features are essential to the core of your product, and which can wait for the next update. As the previous section discussed, you should make your best effort to get the application out ASAP. But don't choose swiftness over correctness. We omitted a pretty important feature from PhotoKast that, in retrospect, we would've put in before the first release.

Categories

When we released version 1 of PhotoKast, it did not have the ability to post photos under categories. All users had to go through all of the different types of photos. This decision was made to preserve the simplicity of the application. If we allowed categories, then there would be an additional step when they uploaded the photo, and there would need to be a screen to select which categories they would like to view from. In addition to the added complexity, this would delay the release by about a week.

That was not a good decision.

What we didn't anticipate was the heated conflict between the different "factions" of the PhotoKast user community. Turns out, people who like viewing pictures of kittens don't usually like the pictures of girls in their bathing suits, and vice versa. Since there are already iPhone apps specifically dedicated to sharing pictures of good looking guys and girls, we assumed that particular demographic was well satiated. *And you know what they say about assumptions.*

Long story short, we pushed out an update and people were well satisfied. Pop quiz: what rule did we ignore when deciding to forego categories in the first release? Making celebrities. A core problem was that some users wanted to be recognized for their artistic shots, but couldn't compete with photos of beautiful people. Now, their photos go directly to people interested in their content, and their photos can now make "Top Kasts" in their own categories. In an update coming soon, we improve upon this concept even further.

Truth v. Reality

Another reason we avoided categories at first was to remove any assumptions that we were promoting certain types of content. The second we define categories, by extension, we acknowledge them. By including "Am I Hot?" categories, we are in a sense promoting that type of content in the application…something we wanted to steer clear of. But in the end, what was meant to protect users only hurt them.

By not defining the categories, PhotoKast didn't explicitly support any particular content type BUT that meant there were no predefined categories a user could subscribe to. Hence, they would have to browse through all of the content, whether they liked it or not. By adding categories for hot guys and gals, we risked looking like we were promoting PhotoKast as a hookup app in order to keep those photos from being broadcasted to people who had no interest in them…a small price to pay.

In the long run, debuting without categories likely cost us at least a ½ star average in the reviews. The reviews trend much higher after the inclusion of that feature. For those that are interested, here are the categories PhotoKast users are subscribed to:

Category	% Users Subscribed
Am I Hot? (Women)	61.7
People	59.1
Other (things, LOL, etc.)	56.2
Artistic	50.2
Activities and Events	49.2
Places	49.0
Animals	46.5
Am I Hot? (Men)	16.7

It's hard to draw a line in the sands of a slippery slope.

"I know it when I see it."
Justice Potter Stewart, Jacobellis v. Ohio 378 U.S. 184 (1964)

Some 50 years ago, a manager of a local art theatre named Nico Jacobellis was convicted and fined $2500 for exhibiting a French film called "The Lovers". His case would find its way to the Supreme Court:

> The U.S. Supreme Court reversed the conviction, ruling that the film was not obscene and hence constitutionally protected. *However, the Court could not agree as to a rationale, yielding four different opinions from the majority, with none garnering the support of more than two justices, as well as two dissenting opinions.* The judgment of the Court was announced by William J. Brennan, but his opinion was joined only by Justice Arthur Goldberg. (Wikipedia; italics mine)

If you've ever had to moderate user generated content, you know exactly how these justices felt. It is almost impossible to define a straightforward standard as to what can be considered offensive and what can be considered safe. It's a slippery slope that users are constantly testing.

Our users might be surprised to find out that we're actually two fairly conservative guys. But we made the decision to allow the community to control the direction of PhotoKast, and not our personal tastes. There is a lot that gets posted that we don't personally agree with. Still, we have to be (and our moderators must be) objective about how we flag photos.

What offensive content standards are used in PhotoKast?

My other half is an obsessed CSI fan, so the upstairs TV is permanently fixed on SPIKE TV where it seems you can find David Caruso's mug reciting his one-liners on an hourly basis (http://www.youtube.com/watch?v=_sarYHi0z948 and http://www.youtube.com/watch?v=glvGfQnx3DI). As luck would have it, I turned the television on to catch the MANswers host ask, "How little can a woman wear in public before it becomes illegal?" How fortuitous.

The law enforcement officer said that it was permissible as long as female nipples, vagina, anus or penis is covered. If any part of the aforementioned is exposed, it is considered illegal. So those are the standards we go by as well. And as cut and dry as that may sound, it still isn't. What if it's a woman wearing a top that is somewhat see through? Then, how see through is flag-able? Trust me when I say that nobody around here wants to spend their time trying to determine whether or not we're looking at a part of a nipple or if it's just heavy pixilation, but these are things that must be done by either you or other moderators. Regardless of how conservatively you flag content, you will no doubt have to endure the submissions from those who define "offensive" differently.

I did a search through the various "public indecency" laws across the country and they are all vague and open to interpretation. This makes it difficult on the people who need to filter content according to strict standards. On PhotoKast, unless a moderator makes a mistake, you should never see a photo that violates the standards listed above. This is not true of some of the other iPhone apps, and a credit to the hard work of the volunteer moderators.

Since Apple doesn't allow offensive content, don't they supply you with the standards to use?

Nope. We've even sent a request for clarification, but have not yet heard back. Since we're being held accountable for filtering out offensive content, we'd love to be able to see a definition of what constitutes offensive content in Apple's eyes.

In their defense, it's probably a smart idea not to provide this standard. It gives them some latitude when reviewing application submissions, and doesn't weigh them down in the quagmire of trying to define (as the justices were unable to do) and enforce a litany of rules and regulations.

Still, as an application provider, we would be upset if our application got pulled for breaking a policy of offensive content that was never defined in the first place.

Require Users to Accept Your Terms & Conditions

When registering a new account, users must accept our terms and conditions, which states (among other things) that it is a user generated content app and, while we try to filter out inappropriate and offensive content, we cannot guarantee it. Also, users are not allowed to post content that they don't own. Be sure that your users understand and agree to your rules before granting them access.

Sometimes, they write the owner's manual.

A few times per day, users will post a PhotoKast manifesto where they try to define the core purpose of the application. We enjoy these kasts, because we've never actually defined it ourselves. We had an idea of how the application should be used, and it's interesting to see users be vocal about what they believe our design objectives were.

What I can say is that it is being used differently by many users than we thought it would.

With the various options for dating and photo swapping applications, we didn't anticipate that there would be that many users who would see PhotoKast as a "hookup" or direct photo exchange application. In retrospect, the fact that the application itself doesn't project the pressure of being a "hookup" application makes it a little more approachable for those who would otherwise be nervous to contribute on apps solely dedicated towards those ends. We get some fantastic artistic photos, beautiful pictures of different places around the world (my favorite category) and some cute animal shots. But the "Am I Hot" section for women is clearly the top visited with 60% of the users subscribed to it (which does mean 40% using the application are not interested in those photos).

Be prepared for the user-base to wrestle the steering wheel away from you. It's how the application takes on a life of its own. We've developed a tool and the people are now using the tool and conforming it as they see fit. While we didn't expect the direction – nor do we personally endorse some of it – we do not intend on trying to conform the submissions to our personal tastes. It is not an application meant to focus on our personal likes, dislikes, beliefs and disbeliefs.

It's rewarding to see reviews that talk about how addictive the application is. It's nice to get the private messages from people who love using PhotoKast. It's a pleasure to see your creation – especially one that took so little time to create – bring joy to many people. On the flip side, it's discouraging to see how a small minority behave with their derogatory comments under the veil of anonymity. Also, I've had to suspend users for posting disturbing content…*very* disturbing content. And so if you are intending to create a social network, understand these issues up front. MySpace was not written to be an enabler to bad behavior, but people use (and will continue to use) it towards their own ends. For better or worse, people define these boundaries differently.

We have enjoyed conversing with the majority of our users, they've been great. But please make sure you're prepared and properly anticipate the scope of running a user generated content application if this is the type of product you intend to create.

Footware, software & customer care.

I'm a Converse guy myself. Sometimes Gola, sometimes Puma. I get the same shoes every time I need a replacement. But I still hate shoe shopping. If only there was one online store where I could find these styles at reasonable prices….

…queue angelic voices…

And then I found Zappos. Within ten minutes, the shoes were ordered and my job was done. A short while later, I get an email telling me they have upgraded me – free of charge – to two day shipping. So I don't have to go to the mall, I get the shoes at a good price, enjoy a 365 day return policy and complimentary two-day shipping? Done deal. It's not like there aren't a billion places to buy shoes on the Internet. But Zappos has used great customer service to their competitive advantage. Since people are used to abysmal treatment, Zappos stands out in a big way. The next time I need shoes, they will be the first place I check.

We try to do a good job with customer service. There are only a couple of us, and we are spending most of our time working on projects for our clients, but many of our users have expressed surprise at how quickly we get back to them. It's a surprise because, unfortunately, they have been groomed by customer service departments that sometimes don't respond at all. Some things to keep in mind:

- **People are patient of you're a good listener.**
  People are willing to forgive bugs and missing features if you're quick to get back to them, give them a time frame for their expectations and show appreciation for their feedback. In fact, this dialog makes them feel a part of the evolution of the product, and it's a great way to recruit evangelists.

- **You can't please them all.**
  So don't base your success on it. When we didn't pre-screen photos, users complained about it. When we started pre-screening, the others would complain. When we didn't allow comment deletion, people wanted it. When we added it, others didn't. You'll just have to make decisions with the best interest of the community at large and be able to explain it to the dissenting factions.

- **Always a handful of people…**
  There are always a handful of people that will puzzle you. Before we instituted categories, one particular user complained about the fact that many of the pictures were of women. He went as far as to give us a "1 out of 5 stars" review in the App Store. So when we added categories, you would expect that he would filter out those photos, right? Nope. He has subscribed to them, and we know this because he leaves comments on the majority of these photos. Despite the negative review, he is also on the application multiple times per day.

- **You're always responsible.**
  When I was in high school (long ago), I started an underground newsletter that – on a monthly basis – would print articles about the dismal state of our school and give progress reports on the different teachers. One of our "reporters" wanted to use profanity in the article, and I objected. He was persistent and I gave in. One of the teachers got a hold of the newsletter and, in the middle of class (and to our great surprise), began praising it. He said he didn't know who wrote it, but he hoped that it would continue…but next time, without the profanity, which he felt undermined our cause. When I approached the reporter and told him this, he shrugged his shoulders and said, "Don't look at me, you're the one who allowed it." And he was right, as the editor I was responsible. And as a software provider, so are you. Your customers don't care that you store your data in Amazon S3 and that it went down, which technically isn't your fault. At the end of the day, it's your product, and the question they are asking is, "does it work?"

- **There are some who will never understand.**
  The iPhone doesn't have a secure way to prevent minors from using applications. As developers, we can only make people aware of the type of content they might see and get their acknowledgement that they've been told. Because of this, we do not allow any form of nudity that would not be publicly acceptable, using the standards mentioned in the moderation section. There are some photos with perceived nudity; in other words, you don't see the illegal portions, but you are fairly confident that if someone else were in the room, they certainly would. The moderators typically stick to the definition, and if they don't see offending parts (regardless of what others in the room may see), they allow it to be posted. We are amazed at the number of requests we get from users who are angry because they cannot publicly broadcast shots of their private parts. We do our best to explain, but some simply aren't convinced that any form of moderation should be done. Stick to your guns. Sure, you'll get some complaints and bad reviews, but you just can't please everyone.

Room for Improvement. There Always Is.

I've been saying a lot of good things about the iPhone. And they're all deserved. Apple has really provided a jumpstart to an industry that was choking on unrealized potential. AT&T also deserves credit for being the carrier willing to make the bold move of giving up some control in order to take the next step. But Apple's endeavor isn't perfect; what is, especially at first? What follows is a list of some of the complaints you'll hear, especially in developer circles. Fortunately, these can all be fixed in due time.

- **"The Gold Rush is over."**
  When the App Store opened, people swarmed it and applications flew off the shelves. By and large, this mad rush has died down considerably. But this doesn't mean that it's still not an effective way to distribute your application.

- **"Application prices are too low."**
  Now, developers who want to sell a lot of units or to get prominently displayed in the App Store are cutting their prices by significant margins. The average cost of iPhone applications are falling, and the negative consequence to this is that other developers must also cut their prices to compete. The lower you go, the less likely you will generate enough revenue to cover your development costs, and so software companies will start abandoning the more complex (and interesting) projects because they can't make the money back. I suppose if there is a positive, then it is the fact that developers must be creative and pay attention to addictive concepts that can be executed in a short period of time. As we (Ten23) are breaking ground on our new social networking app, we are also dismayed at the low prices and challenge of making our money back. But it does force us to really identify what addictive features we can add that users would still be willing to pay for. Free markets tend to work themselves out.

- **"The Apple application review process is too subjective."**
  There have been a number of developers who have complained about Apple wielding such tight control over the review process. They are afraid that they will spend time developing an application that will ultimately get rejected, something that has happened to a few (including an app that makes fart noises, which has recently been re-reviewed, approved, and now sits atop as one of the bestselling titles). To be honest, this was a concern for us too. In our experience – SO FAR – Apple has been great with our submissions. Almost all of our application and update submissions have received a response within 4 days. Check out the developer forums for posts from other developers and the range of experiences they have encountered.

- **"The App Review duration is unknown."**
  As I had mentioned, Apple has been good to us, responding to our submissions with an approval (or rejection) within 4 days each time. Others have had to wait longer – in some cases, much longer – to receive a response. You can hear from many of the developers experiencing the delays in the Apple Developer Forums.

- **"The App Store could do a better job at navigation and promotion."**
  The App Store is an incredible contribution that has wonderfully impacted the lives of mobile developers. In the beginning, it was easy to navigate and find applications of interest. However, as more and more applications are added, it is becoming increasingly difficult to get your own titles noticed. While they enjoy some time at the top of the lists, new releases quickly push your

application to the bottom. It would be nice to be able to easily access applications per category based on their popularity, review counts and ratings, as well as to rotate the top titles in a featured page that is more dynamic than the one currently available. I suspect that these types of changes will be introduced as the App Store continues to evolve.

- **"Yet another new mobile development platform."**
  It's not an easy task to create a mobile application that will run on many devices. For most mobile phones, you will need to use Java ME (and even within this group, there is much fragmentation – for example, BlackBerry devices are Java ME based, but offer specialized classes that better integrate with their device capabilities). For other phones, you may need to target other operating systems like Windows Mobile. And now, there is a new platform to learn for the iPhone. Objective-C and Cocoa Touch are large enough departures from the others to give developers fits. But with that said, it is a very capable and stable platform to develop for.

- **"Apple has too much control."**
  Apple controls the submission process, and who gets on board or not. They control the terms and conditions for the App Store and use of the SDK. In short, they can delay your submission review indefinitely, prevent you from getting approved, or pull your once approved app from the store without notice. All of these things cause developers great concern. Perhaps the best consolation is that – while there have been incidences – things have run quite smoothly. In fact, applications that were once rejected are now getting approved.

In Conclusion

Bringing a product to fruition is quite an accomplishment. Even for nominal software, there are a series of hurdles that must be overcome for your application to see the light of day. The quality of the application is usually a product of how well you've overcome each hurdle. And while there will be many, there are six main sections in the pipeline:

Concept	Do you have an interesting concept for a mobile application? Can you describe it in a sentence or two? (ex, "PhotoKast is a photo sharing application that lets users track the worldwide popularity of their pictures.") Can the concept be properly ported to the mobile device, and will it be addictive? Be careful not to conceive of an application that has too much to it; if the concept doesn't work in its simplest form, loading it up with bells and whistles is like (to use a well worn expression) putting lipstick on a pig.
Design	Historically, this has been an often overlooked discipline. Many who lead software teams would rather see their team members pounding away at the keyboard rather than staring out of windows; but sometimes, that's what they need. Properly allocating time to design will save you time later; through proper design, construct a clear blueprint that will ensure you won't have to readapt programming efforts to a constantly changing directions. Spend time sketching your concept in the form of user interfaces.
Development	Development (and debugging) is a very important and difficult part of the process. Obviously, you don't want buggy software, and you'll need a team that understands how to properly code and test their software. I've seen so many people fail at this point. If you do not have an in-house team to do your development, be especially sure that your concepts and designs are mature – never leave it up to the development team to fill in the void. Obviously, you will need a team that excels where you are weak (concept, design, coding, etc).
Distribution	What is your plan for distributing the application? If you're writing an iPhone application, the App Store is your distribution vehicle. How will you reach your intended audience- do you have a social media strategy?
Monetization	Some applications are loss leaders for other services or brands. Most hope to generate revenue from their software, usually through a one-time cost, reoccurring subscription fees or ad revenue. Make sure you have already defined how you will make money, and also make sure that the proper hooks have been built into the application before its release. Be realistic about how much you expect to make and how much your operating expenses will be.
Maintenance	Maintenance is a very important part of your application. This section includes software maintenance such as new versions, bug fixes and software updates; it also includes proper customer service.

So, there you have it. I suspect if you've gotten to this point, something has kept your attention. Or, you've just skipped here to see how the story ends. There are many more tips, tricks and suggestions that could be offered, but that would make this a book and a full time job. I'll leave you with some PhotoKast stats (some of which have been presented throughout this document) and some good resources for you to check out.

PhotoKast Stats

- 2+ million photos served a week (only counting photos served for voting; not browsing of Top Kasts, etc.)
- PhotoKast PRO users can send unlimited private messages (with photo attachments) to one another. There are 4X as many private messages w/ photos as there are kasts.
- ~20% are removed for not meeting the public posting decency standards.
- PhotoKast PRO in top 10 paid social networking applications during first month, even with a free version available.

Resources

We were not asked to list any of the following entries; they are resources that we use and think that you might find interesting as well.

Apple's iPhone Developer Program
If you're serious about developing iPhone apps, you'll need to sign up for this.
http://developer.apple.com/iPhone/program/

Apple iPhone Dev Center
http://developer.apple.com/iphone/

Inside iPhone
A great resource for iPhone developers
http://blogs.oreilly.com/iphone/blog/

Furbo.org
A useful blog written by iPhone developer Craig Hockenberry.

http://furbo.org

iPhone Development
Jeff LaMarche's development blog.
http://iphonedevelopment.blogspot.com/

iPhone Application Programming – CS 193P Stanford
GREAT resource on iPhone development.
http://www.stanford.edu/class/cs193p/cgi-bin/index.php

Beginning iPhone Development: Exploring the iPhone SDK
Dave Mark, Jeff LaMarche
Great book for those getting their feet wet with iPhone development.

iPhone SDK Application Development
Jonathan Zdziarski
Good, new book on SDK development

iPhone in Action: Introduction to Web and SDK Development
Christopher Allen and Shannon Appelcline
Highly recommended, great overview on both iPhone web and SDK development

Programming in Objective-C 2.0 (2nd Edition)
Stephen G. Kochan
Good overview of the Objective-C language.

Kicking Butt with MIDP and MSA
Jonathan Knudsen

Good book for those Java ME developers out there (not iPhone related)

A Whole New Mind
Daniel H. Pink
A good read.

Professional Android Development
Reto Meier
Interested in Android? Check this book out.

NAVTEQ
Mapping data for those interested in LBS mobile applications
http://navteq.com

deCarta
Geospatial software solutions and expertise for location-based applications.
http://www.decarta.com

RIM BlackBerry Developers
Working with BlackBerry devices?
http://na.blackberry.com/eng/developers/

Forum Nokia
Working with Nokia devices?
http://www.forum.nokia.com/

Contact

If you'd like to contact me, please do so at my email address: cheiser@ten23software.com

* Are you interested in using technology – particularly web and mobile technology – to help NPOs, etc? If so, I'd like to hear from you.