ROSE-HULMAN INSTITUTE OF TECHNOLOGY **CSSE483 Android Application Development**

Term Project Specifications

Every student in this course must complete a term project. What opportunities do you get from this experience?

1. Working on an Android application independently (apart from lab instructions).
2. Exploring parts of the Android API that we wouldn't have time to cover in class.
3. Applying knowledge of the process of software development to a new application.
4. Having fun developing an app that interests you.
5. Working with a classmate to make something better than either could make alone.
6. Satisfaction in creating something **useful and complex enough to be posted** on Google Play.

Grading

Grades will be broken down as follows.

Milestone	**Weight**	**How assessed?**
Brainstorm: Three ideas	3	Document
Initial Project Planning Doc	3	Document
Mocks	5	Mockups (screen layouts and navigation between screens)
Design	5	Document
Sprint 1	12	Code demo to instructor or assistant
Sprint 2	12	Code demo to instructor or assistant
Final demonstration/presentation	50	Video demo, code submission
Documentation	10	Document: technical discussion of new feature
Total:	**100**	

Project Requirements

For an A,

- The app must be significantly more complex than a lab.
- The UI must be polished (colors/fonts, easy to use)
- The UI must be sufficiently complex (including appropriate interfaces, like calendar chooser), typically 5-15 screens or dialogs
- Navigation must be easy to follow
- The data you store in Firestore must be reasonably complex (more than one table; show backend in demo video)
- You must include some significant functionality beyond what you are taught in class.
- You must attempt to meet an actual need
- the app should take advantage of being on a mobile device (because small/accessible/always on, because of sensors, because of the touch screen, or because of connectivity) rather than on a laptop or desktop.

Milestones

0. Brainstorming (in class or in discussion forum)

Finish the statement, “If only there were an app that …” Consider some of the features that make for good apps: making use of their small size, their sensors, and their internet connectivity.

1. Three product idea sheets

The goal of this milestone is for you to practice identifying a real need that an app could solve, and then propose a solution. You’ll also research to see how this need is currently being met by other products. By all means, look at other apps just to see what can be done in Android. In addition to looking at sites like Google Play and the app store (for iOS), you may find other sites like appcrawlr to be helpful.

Note: app ideas may not encourage illegal or unethical activity. This includes alcohol-related apps or those with sexually-explicit content, given that your peers are largely underage.

If you and a classmate have already agreed to work together on the project, you may submit one set of three ideas per pair[1]. Otherwise, you must submit one set of three ideas per person.

a. Open the template below.

   https://docs.google.com/document/d/1Ssek5fupoboshHoPWMa9ce6w9jE01w6GysLSu5V02KU/edit

b. File > make a copy, and file > rename the template as *Android Project Idea Sheets - Name1 Name2*.

c. Use the template to write your ideas. The template is **3 pages: one per idea you have*!**

   The template has additional specifications. Read them!

d. Sharing:
e. Make it public. (Otherwise I can’t grade it. :))
f. Submit the link to it to this form.
g. In a couple weeks, the link will go into the Rose Hulman Project Vault (more below).

I will post your ideas on the course website for others to see.

[1] See the end of milestone 2 for a discussion of teams.

Others' ideas with known value

I know people who would LOVE to have you develop an app for them! The great thing about working on one of these is that you KNOW that there is a need for the app.

1. The Haute (Mark Gibson, Envisionary Media)
2. Bible MishMash (Janice Fenn, former head of the RHIT Center for Diversity)
3. PrayerNet (Dr. Boutell for a friend of his who runs a ministry)

*If you are seriously considering working on one of these apps, you may choose one as one of your ideas (so only need to do 2 others) and we'll talk about potential next steps.

2. Initial Project Planning Doc

The goal of this milestone is for you to choose a partner and project, to add more detail, and to respond to feedback from the idea sheets, if appropriate.

Your ideas must meet the requirements above, so **convince** me that each should be a mobile app, that it meets a real need, and that each would be complex enough to require multiple screens.

Keep in mind that the grade for your final submission will include components for applying what you learned (course big idea 1) and for daring to do things beyond what we covered in class (course big idea 2). So make sure that your proposed project isn't too easy.

To give you an idea, an appropriate size for a project is typically 1-2K lines of code...long enough to require you to dig deeply to learn on your own, and complex enough to publish on Google Play, even if it is not polished enough yet to do so.

For example, here are the features you should plan to have:

1. Storage of your data in the cloud (via Firebase)
    a. Enough data to store that it is interesting (I expect the complexity to vary between apps, of course. If your data is super-simple, you should have something else complex to balance it out.)
2. Cloud authentication (at least one type)
3. A visually pleasing interface: clean layout, colors, icons.
4. An easy-to-use interface with intuitive ways to switch between your several fragments or activities - see the Activity layouts shown when you create a new project for ideas.
5. Some feature that goes beyond what we learned in class. (Easy to do if you plan early.)

Write up your idea in a formal proposal. The format of the document is:

1. Project Title
2. Names of students on project
3. Summary of the project (2 paragraphs)
4. Preliminary list of features.

5. A draft screen layout or two to get you thinking about layout. The layout can be handwritten/scanned for this milestone. Try to capture your most representative or most important screen or two.

You are expected to work with a partner. In rare cases, an individual may be allowed to work alone if that individual petitions me in writing with a very good reason. However, **an individual must still complete as much work as a team of two**. Also in rare cases, a team of three can work together, but the app must be at least 50% more complex than one developed by a pair.

Submit the link to it to this form. The next 2 milestones will build on your initial project planning doc, so you'll be updating your document but you don't need to re-submit the link. Instead, next week you'll move it to the Project Vault.

3. Annotated Mocks (Screen Layouts) and User Stories

Summary: Wireframe your electronic screen layouts using a mockup tool called figma and explain the different actions via user stories. **Add** them to your initial project planning doc from the previous milestone and add the planning doc to the Project Vault.

For a good example of the level of professionalism expected, please read

http://developer.android.com/training/design-navigation/wireframing.html.

Read and follow Google's guidelines for designing effective navigation at:

http://developer.android.com/training/design-navigation/index.html

You don't need to follow their format exactly (with arrows, etc), but you should include all your screens and make the navigation pattern very clear using Figma's prototyping functionality either visually or with text.

I require Figma as the prototyping tool (intro tutorial).

Start by making a free student account, so that you can get sharing/team access for free. :)
https://help.figma.com/faqs/setting-up-your-student-account

Then make a draft project in Figma, and either make it publicly viewable or add me (boutell@gmail.com) to the project so I can view and give feedback.

Since Figma shows navigation interactively (short video about how to set up prototyping), you can just include a link to your mock in your project planning document. (Or if you preferred, you could additionally grab screenshots of your mockups and add them to your document and use visuals or words to clearly show navigation.)

This is also the natural time for you to develop your list of requirements into a list of *user stories*. ("When the user does X, the program responds by doing Y…"). Some of these will already be clear from the mocks (like "When the user clicks the login button, it loads the LoginFragment where they can enter their RHIT credentials") so you don't need to add anything to them. But others won't be clear (like, what does swiping the screen do?) and you have two choices: The nicest way to do this is usually just to **add captions** to your mock. Or if you need to write more, or if you don't like captions, you can just include a list of user stories in your project planning doc. No problem.

Make sure as well that you show in your mocks **how you will login** to your app, as all apps have user accounts and authentication.

Rubric (5 points):

> **5 (excellent)**: Anticipated all screens needed, including login, mocks done using a tool, mocks clearly show the UI elements and all navigation between elements. All user stories shown in annotations or in a list.
> **4 (good)**: Has most screens, most elements present, but leaves the reader with questions about usability, navigation, or user stories.
> **3 (passing)**: Has the most important screens, UI elements mostly present, navigation unclear, some user stories unclear.

Submission:

1. Add a link to your **Figma project to your project planning doc**.

2. Go to the **Rose-Hulman Project Vault**. This is a super-cool web app that you'll be using for the rest of the term to submit all your project work! You have a page there already reserved for you. To find it, just click your term's link (link for Winter 19-20) and find and click into your team's project. (Alternately, you can go to the Rose-Hulman Project Vault main site and find your project by choosing "my projects" in the side nav bar.)

3. Once you are in your project, press Edit in the toolbar and change at least your title if needed. You can also add a subtitle and description. Then scroll down to find the places to copy in your links to your **project planning doc** and your **three ideas doc**. (You can choose which student's 3 ideas doc to link to, if you joined after you each created 3 ideas.) **Click Save** in the Toolbar when done. Then click back to check that your project is updated. While there, click into and edit your user profile with as much info as you have plus a photo (**saving** when done).

4. Your project page is yours, so you can start to add the other required information at any time this term.

4. Design and Admin

Do the following:

1. Create a new section in your project doc called **Database Model Object Schema** that will show how you plan to organize your data in the database. What objects do you have? What properties do they have? How are they related? Don't forget to model your users, since most apps will have data that is stored on a per-user basis. For example, you could use a UML class diagram for this, with a box for each object you'll store, listing all the fields and types within each, plus arrows between boxes showing the relations. Or you could use another format that you prefer - I'm less worried about the format than I am about you thinking early about the data and its complexity. This will turn directly into data to be stored in Firebase, as you'll learn in week 5. You just need to model the **data, not the UI** - the mocks were sufficient for planning the UI.
2. Your goals for sprint 1[2]. Add this to a separate "sprint" google doc, with view rights to anyone with the link, which you'll add to the RHIT Project Vault (like you did your planning doc in the last milestone). For each task, note which partner is taking the lead on it, even if you plan to pair-program it.
3. A link to your github repo. You just need to create the repo at this point. Public repos are fine. (If you feel the need for a private repo, I think Github gives them for free to student accounts, but you may have to request one. And if you do, please give me access - my github username is mboutell.) Add this link to the RHIT Project Vault also.
4. If you have anything missing from your Vault page from the last milestone (project planning doc, 3 ideas) or your personal profile (photo and notes about yourself, which are both required for full credit), please do so now.

This design and setup is for you, so you don't have to go overboard, but your attention to design up front will have big payoffs later while coding. Doing the administrative work this week will help you to hit the ground running with Sprint 1. Make sure you discuss how you will collaborate with your teammate. Pair programming is often best if you are in the same location.

5. Sprint 0.5

You must have made some significant progress during the first week of sprint 1. I will check your progress in your repo. I am checking to see that each of you made several commits. Consistent progress through the week, evidenced by your commits, is best. (2 of the 12 points for Sprint 1).

5. Sprint 1

The purpose of this sprint is to get off to a good start on your project. Most students prefer to

---

[2] If you aren't familiar with the idea of *sprints*, they are short (1-2 week) milestones with very clear goals for which features should be complete. You'll do 3 sprints, with each member continually pushing changes to github so that you can show steady progress on each one. The first two sprints are 2 weeks each and are even broken down to one-week "half-sprints" to reflect the expectation that **each of you are committing code each week**.

start with the UI and delay the backend until you learn how to use Firebase. This gives you significant progress and a sense of accomplishment early. But don't be tempted to wait until sprint 2 or 3 to try out new technical areas. **You will meet with your instructor or an assistant to demo what you have done**, and to discuss what you plan to complete during sprint 2.

Deliverables:

1. Code (will demo on your laptop or device. If you are using a repository like git, please share with the instructor as well)
2. Your revised sprint document, completed **before** the meeting (at least 2 hours before). The only thing new in it is a brief status update, including how you accomplished each of your sprint 1 goals. Update who ended up leading each task if it changed. then add your list of goals for sprint 2, with who is leading each.

5. Sprint 1.5

You must have made some significant progress during the first week of sprint 2. I will check your progress in your repo. (2 of the 12 points for Sprint 2).

6. Sprint 2 and 3

Like the previous meeting, your meeting will consist of a demo of your project, and your revised design document, updated before the meeting. Again, explain what you completed relative to what you had planned to do, with who led each, and include goals for what you plan to complete during the next sprint. (There will be a sprint 3, but no review at the end, since you will do the final presentation and submission at the end of sprint 3.)

7. Demonstration/Presentation

You will create a video demonstration of your app for your classmates, me, and other faculty/staff to evaluate. Please follow these guidelines:

1. Speak loudly and clearly.
2. Start by clearly stating each of your names and the title of your project.
3. Discuss briefly your reasons for choosing to develop this app: what need are you trying to meet? How are you hoping to create value through the app?
4. Walk through your app's features. Present as if you are talking to someone who has never used your app (since your classmates won't have used it).
5. Briefly discuss one interesting technical feature of your app that used concepts beyond the material covered in class (as required in the spec). Stay at a concept level only, since this is a teaser to get classmates to check out your technical documentation (below).
6. Submission: for this one, please upload it to **YouTube.** For access, you may choose **unlisted** (if you just want those with the link to see it) or public (if you want anyone and everyone to see it). Private won't work, obviously.
7. Submit the YouTube ID to the RHIT Project Vault (it gives an example of the ID as a hint on your project page - it's part of the URL).

    a. Also, this is your chance to make sure that your submission to the Vault is complete, up-to-date, and finalized: add a few screenshots, your profile and pic, and write a few words about the app and yourselves as developers.
    b. Link to Winter 2017 projects for examples of completed submissions
8. The length of the video may vary. I expect most will be in the 5 minute range, if you include good detail and plan well.
9. Options for recording: (1) Use the emulator and use screen recording software that captures voice. See here for past CSSE483 students' favorites. (2) Use a videocamera (even your laptop webcam) and record yourselves and then zooming in on the device. Nicely personal, but be careful about glare on your phone/tablet screen from lighting.(4) If you like other options, please post them to piazza! Your classmates and I would love to hear from you.
10. (Summer sections **only**: you may choose to have one teammate record this presentation solo, since it would be difficult to coordinate if you are in different locations. Of course, you should plan the content together.)

8. Evaluations of Other Projects

You will evaluate the quality of the other teams' projects using a private google form using a link that I'll add to each project (green button). We'll use this in part to choose the "Most Valuable Project" awards. And if you really like an app, please write a short public review (blue button) too!

(Face-face class only: we'll present these in class on Friday of 10th week; if you think your app might be one of the best, please be ready to present in class that day!)

9. Technical Tutorial

You were required to use some feature in Android beyond what we did in class. Here you will provide a short tutorial of that feature: your target audience is students completing a similar project in the future, so make it helpful. Don't be vague: write a brief (~2 pages, but that's pretty flexible) tutorial including and explaining the *key code snippets*. Ideally, try to pick something that you don't think many other teams did. **Use a google doc with edit rights for me (boutell@gmail.com) and rights to view for anyone with the link** and submit the link to the RHIT Project Vault (if you edit your project, it's the "technical doc" field near the bottom). One note: I expect that you learned this info from other resources - your job here is to summarize and to clarify any "gotchas". **Please cite your sources (a link to them is fine).** *Plagiarized info just copied directly from other sources without citation or without any summarization is unhelpful, unacceptable, and will result in a grade of 0%.*

10. Final Code Submission

Past students have asked for their classmates' code to learn from. Please make sure your final submission is pushed to github.

11. Partner Evaluations

You will complete the project by evaluating your performance on this project and your partner's performance on the project. This is a Moodle survey.

Grading guidelines

All work will be graded on the following 10-point scale:
10 = Excellent Team exceeded project goals and expectations
9 = Very Good Team is in good shape and doing very well
8 = Satisfactory Team has met the requested ideas at a satisfactory level
7 = Ordinary Team is okay but isn't at the appropriate project status level
6 = Marginal Team is not in real trouble yet but behind target status
5 = Deficient Team is behind project goals
4 = Unsatisfactory Team has little to show for the time allotted
3 = Superficial Team has thrown something together at the last minute
2 = Gave Excuses Team had nothing to give other than an excuse
1 = Dim-Witted Team had nothing to give and couldn't think up an excuse
0 = Not Present Team not present for meeting

Note that a 9, not 10, is the typical grade for solid, very good work. You need to go above and beyond to each a 10 (100%).

Furthermore, the grade on the final submission will be capped at an 7 (so 35/50) if it only applies ideas introduced in class. Grades of Satisfactory, Very Good or Excellent are only for teams also successfully demonstrating ideas **beyond** those learned in class.

Anything in this document is subject to change at the instructor's discretion.