CSE 6324 – Advanced Topics in Software Engineering

Agile - Scrum

Spring 2020 – Term Project Options

Term Project Objective:

Design and develop an android application using an agile software engineering methodology Scrum. This application will be developed in an incremental manor using all agile practices such as pair programming, refactoring, test-first and active client involvement.

Project Requirements:

- Please pick the Android project (listed at the end) as your term project. Your software should be an android application that runs on the Google Android platform.
- Please keep in mind that CSE 6324 is an agile class and not an android teaching class. Each team is responsible to learn the basics of android programming. A number of great links provided at the end of this document that helps you start with basics of android.
- Each team will demo your completed story using the available DocCam in the classroom or (preferably) using a remote control access (instructions will be provided later). You may also demo via an Android emulator if you prefer. If you use emulator mode, you must provide the necessary software required to run/test your application.
- You may select any android development environment, however one of the latest versions are preferred (i.e. KitKat, Lollipop, or Marshmallow). Android history can be found at: https://www.android.com/history/#/donut
- You must develop the project yourself. It is not enough to just copy paste a project from somewhere else. This does not mean you cannot reuse any existing code. Many software engineering products build on some existing code. But it is important that you clearly document which parts are yours and which parts you reuse from another source. If in doubt, please run it by me and our GTA first!
- Project will be completed in teams, no individual work is accepted. All team members are expected to put equal time in handling the allocated tasks. In case of any friction or miscommunication, please inform me and GTA immediately. I will conduct a peer-review later in the semester to evaluate all team members' equal participation.

- No copyright violations for any part of "reused" code, if any. Provide references when using other people's work.

Agile organization and activities:

Each team consists of 7 members with revolving roles as follows:

- We will use Scrum methodology in this class. Each increment (i.e. sprint) in scrum could be 2 to 4 weeks. Please pick 2 weeks for your sprints.
- Project is divided into a number of increments, i.e. Sprints, each to be completed in no more than 2 weeks. If a story is going to take longer than 2 weeks, break it down further.
- Since the semester is finite and we will end during the first week in May, we cannot complete more than 5 sprints (5 x 2 = 10 weeks).
- For each "sprint", one member will play the role of the "ScrumMaster and Client", the other members will be "developers" working in pairs.
- The "Client" role includes: help identify and prioritize increments and stories, participate in writing test cases prior to starting implementation (i.e. test-first strategy), make daily builds and, most importantly
- The ScrumMaster role manages all the technical details of an increment (as well as playing the "client role"). ScrumMaster position will be rotated so all member can practice. ScrumMaster may NOT develop code.
- The "Developer" role includes: implementing the required activities including but not limited to: writing requirements, design & coding, unit testing, writing test cases, integration and testing. Developers will work in pairs at all times.
- "Pairs" will be rotating as sprints change (i.e. no two people will be paired more than once)
- "Daily Standup Meetings, DSM, also referred to as daily scrum": Normally, all team members will present their work on a daily basis while standing. In this class, since we meet only twice a week, we will have DSM every Thursday (10 minutes per team total).
- DSM format and structure: ScrumMaster will start the meeting, and passes the turn to pairs. ScrumMaster will summarize and end the meeting. Each member will go over:

    - What did they finish since last DSM?
    - What are they going to work on next?
    - Any issues?

- Test-first" strategy: Before any development, test cases must be written of how to test the functionality. Test cases will be discussed briefly during DSMs. The exact format of test cases will be discussed later in class. In ideal situations, Junit should be used.

- "Refactoring": Sufficient time will be devoted to refactoring (i.e. continuous optimization and simplification) and documenting the changes due to refactoring.

General comments:

- All team members must participate equally in all stages of the project. A peer review will be conducted to validate equal participation.
- Establish a weekly meeting with your team members. Typically a one-hour session is sufficient at the start. Make sure all members can attend and are committed to this beyond class period.
- Only ScrumMaster will communicate with me and class GTA for project related issues.
- Bonus: 5 points for the best class project.

Scrum Progress & delivery:

Your project will be delivered as follows:

Deliverable	Due date
Project backlog – hardcopy; Team present in class	Feb. 6, 2020
Sprint 1 – Presentation/ Retrospective/Summary report	Feb. 10-21
Sprint 2 – Presentation/ Retrospective/Summary report	Feb. 24 - March 6
Spring break – no class	March 9-13
Material Review (3/17) and Test 1(3/19)	March 19 – Test 1
Sprint 3 – Presentation/ Retrospective/Summary report	March 23-Aril 3
Sprint 4 – Presentation/ Retrospective/Summary report	April 6-17
Sprint 5 – Complete Project Release/Delivery on 4/27 Presentation/ Retrospective/Summary report	April 20-May 1
Project Delivery & Closure (5/5) Test 2 (5/7)	May 5 – Project Closure May 7 – Test 2

- Project initiation
  - Define stories and sprints (5)
  - Finalize story lengths (man/days) using poker planning approach
  - Prioritize stories and define sprints (which stories will be done in which sprint; sprints could be 2 to 4 weeks in Scrum, we will pick 2-week sprints)
  - Create project backlog **(first deliverable for this class, due Feb. 6st, Submit project backlog and Burndown chart, present details as a team in class on Feb. 6th)**
  - Phase I: decide which sprints will be completed by end of this semester
  - Phase II: decide what should go in phase II (i.e. hopefully nothing, only future expansions)

- Sprint planning for Phase I (sprint 1-5)

- Sprints 1-5:
  - Develop Test cases using designated template and present 1st Thursday
  - Design & Develop & Test
  - Release: Demo app using Remote control the **2nd** Thursday
  - Submit updated Project backlog, Burndown chart and Retrospective by 11:59 PM on Friday via e-mail (first one before start of Sprint 1, i.e. Friday Feb.6th; then once every two weeks, at the end of each Sprint)
  - On each Thursday, please hand-in two hard copies of your test cases (week 1) and stories completed (week 2) just before you start your DSM.

- Project Delivery and Closure:
  - All items listed for under Sprint 1-5 listed above plus:
  - Final Release/Delivery (Delivery instructions will be provided later)
  - Update project backlog (Anything left in backlog, will be phase II)
  - Phase I completion – Final report, demo and project closure "Release"
  - Phase II planning (future work – hopefully none, i.e. Stories that are not completed in Phase I)

Project: ServeMe System (SMS):

Develop an android application to help users initiate "service request" for their home or small business. This will help users (i.e. customers that require service) to be put in contact with service providers. You will make money by collecting commission from each service request fulfilled by the service provider. The categories of services (displayed graphically by an icon) can be expanded to cover more service areas in the future. Initially the listed services should be supported. The system should minimally include the following components:

1. Login & Registration: Customers (i.e. Service Requesters) can optionally register and login to the system. They can also order place a Service Request (SR) without registration and login. If they choose to register and login, they could get points, can review Service Providers quality of Service, could get discounts, and receive helpful tips and updates. To register, they must provide full name, email, and phone number.
2. Setup: Setup control fields such as: Sounds **on**/off, receive communication **yes**/no, use points toward a service yes/**no**. feel free to include other parameters. The Bold charters will be the default.
3. Service Categories: The following areas of service should be presented in the app. For each category of service area below (i.e. similar to Uber for rides):
    - Appliances
    - Electrical
    - Plumbing
    - Home Cleaning
    - Tutoring
    - Packaging and Moving
    - Computer Repair
    - Home Repair and Painting
    - Pest Control
4. Register to become an approved vendor for one or more of the above service categories (i.e. Service Provider). You must provide contact info including phone, address and e-mail. You can optionally provide sample rates (e.g. how much to fix something, or per hour of labor). You must also agree to pay a small portion of money received (e.g. 20%) as fee to the app holder
5. Place a Service Request (by Customers): Select a Service category and place an order soliciting for bid by the service providers. Finalize your selection after reviewing all bids.
6. Cancellation and Change: Allow customers to cancel or change service dates. You can specify in setup that you lose points if you cancel within the last x (e.g. 24) hours
7. Accept a Service Request (by service provider)): Provide bids for Requested services to customers
8. Payments: Handle money paid by the customers, received by the service provider. You, the owner of app, will receive a commission (e.g. 20%) of the service fee.

9. Review and Rating (by customers): Use Stars (one to five stars) to rate the service received, and allow to enter for a short comment to help future users
10. Order History: Display the history of service requested for a given customer.
11. Search: Ability to search by for a service provider by address, name, star rating, etc.

Android: Resources and references you might find helpful:

All Android related questions should be sent to the class GTA first. It is not required to now android up front however if you want to have a head start, the links below, in the order listed, can be of great help.

Getting started with Android Programming: http://developer.android.com/training/index.html

Android SDK: http://developer.android.com/sdk/index.html

Android Development Tools (ADT): http://developer.android.com/tools/sdk/eclipse-adt.html

Android Development Tutorial: http://www.vogella.com/articles/Android/article.html

Android Studio Download: http://developer.android.com/tools/studio/index.html

Eclipse Downloads: http://www.eclipse.org/downloads/