SWEN 444 Human Centered Requirements and Design

Project Breakdown

Team Status Reports: (starting in Week 2)

Your team will **report weekly** project status to your instructor, and as you wish, capture other project related information. The weekly status report should contain:

- A list of what **tasks/features** were **accomplished** for the week.
- **Who** worked on each task/feature and the **numbers of hours** each team member contributed to the work. (This information will be used to help evaluate individual team participation.)
- A list of what **tasks/features** are **planned** for the **next week**.
- A list of project related **risks**, **challenges, or issues** that you need to share
- You may choose the reporting tool; e.g., a blog post, Trello, shared document, etc. as long as the instructor has access to it.
- The status report will be checked as part of each project deliverable evaluation.

Note: All due dates are listed on myCourses.

Deliverable 0: Project Startup

Your team needs to submit a list of **three project ideas** (in the shared Google spreadsheet). **With instructor collaboration**, you will then pick a project topic from one of the three ideas. The instructor may assign problem domain themes that require some preliminary research to become familiar with the problem domain and/or find users with domain knowledge. The goal is variety in class projects, but efforts will be made to select a project from your list of preferences. You will need to select an application platform from the categories below. Pure browser based web applications are acceptable as long as they are suitably interactive, comprehensive in scope, and original.

You will be building an **interactive prototype** of your design. Your proposal should include a **discussion** of **how** you will **develop the prototype** on the chosen platform. Prototype development should be minimalist, we want to focus on the interactive design not software design. **DO NOT pick a platform that you are not familiar with!**

NOTE: As an alternative to programming the prototype, you may choose to use a collaborative prototyping tool. An example is Mockplus (https://www.mockplus.com/).

Platform Categories:

- **Web Browser** – develop a web application – it needs to be substantively interactive and innovative.
- **Android Platform**: Develop a native Android app (you need to own an Android phone or tablet).
- **PC Desktop Platform**: Develop some application native to a PC, or as a PC based simulation of the interface of an external or embedded device such as by example POS terminals, consumer kiosks (e.g., airport check-in kiosk), media players, home security system, software engineering tools ...
- **Other platforms** – other hardware/software platforms may be used with **instructor approval.** You need to supply the platform hardware and software. Examples might be Microsoft Kinect to develop a non-traditional application based on the use of gestures and spoken commands, or Raspberry Pi for some custom application idea.

The **final project** selection will be made **in discussion with the instructor**. Once you have selected the project idea, write a description for it (**system concept statement**) and submit it to the appropriate **Project Dropbox " Deliverable 0: Project Startup"**.

Choose a **team name**. The default will be "Team n" where n is the group number.

Once your project is known, **recruit four users** from the **target population** for your system. None of your users should be enrolled in SWEN-444. These individuals must be willing to answer questions during requirements elicitation, and eventually participate in user testing.

Deliverable 1: Contextual Inquiry and Analysis

1. Review the **system concept statement**. Use it for interviewing and observation.
2. Interview/observe your four candidate users. **Document the interview and observation process** - who, what questions, what notes.
3. Identify and document **work roles.** Profile each role in a table to include name, personal characteristics, and abilities.
4. Create a big picture system **flow model diagram**.
5. Use the **requirements template**.
6. **Team status report**.

Deliverable 2: Work Activity Affinity Diagram and Requirements

1. Each team member should independently synthesize **work activity notes** from the raw interview notes. Then as a team create a work **activity affinity diagram** using the work activity notes using the process discussed in class. Submit a (high resolution) **photo of the completed WAAD** and a short **synopsis** of the team's reflection on the process experience.
2. Derive **interactive design requirements** (not software requirements) by walking the WAAD one work activity note at a time. What **user needs** are implied by the work

activity note? **Translate** each **user need** into one or more **interactive design requirements**.
3. Document relevant quantified **usability requirements** for learnability, memorability, efficiency, understandability, and satisfaction.
4. Document the requirements using the **requirements template**.
5. **Team status report**.

Deliverable 3: Design Models

1. Review and refine primary work (user) roles and the work flow diagram. Show interconnections among components of the work domain. Show work flow, information flow, and all communications among the components.
2. Make a social model diagram. Identify active entities and represent as nodes. Show norms of behavior, concerns of individuals in specific work roles, influences, feelings, and environmental factors.
3. **For each primary role** perform HTA for **relevant and significant tasks**. You should model at a **minimum five tasks in total**. Document in the **requirements template** each HTA using the **descriptive format** (the graphical format is not required).
4. Write **usage scenarios**, as task interaction models, for **five** features supported by the system. Describe key usage situations happening over time for specific people with work goals.
5. **Team status report.**

Deliverable 4: Conceptual and Intermediate Design

1. Construct a **persona** for one work role.
2. Hold a team ideation session to produce a conceptual design. The design should be represented as **sketches** and **possible features.**
3. Create **storyboards** (ecological, interaction, and emotional perspectives) for your design. What **mental models and metaphors** are represented?
4. Refine the design as a **wireframe** in preparation for in-class cognitive walkthrough evaluation.
5. Submit the persona description, sketches, storyboards, metaphors/mental models considered, and wireframes. Use the **Interactive Design template**.
6. **Team status report**.

Deliverable 5: Detailed Design

Refine the intermediate design to provide a more detailed design prototype. Incorporate feedback from the cognitive walkthrough. Apply design guidelines. Record the rationale for your design decisions.

This evolutionary prototype should be:

- **High fidelity in look and feel**. Use this prototype to explore the graphic design of your final implementation. Lay out screens as you want them to appear in your final

implementation. Make choices about colors, fonts, alignment, icons, and white space. Your prototype need not be pixel-for-pixel identical to your final implementation.
- **Medium fidelity in breadth and depth**. Your prototype should be able to handle at least the **five tasks** you described in your **task analysis**.
- **Be prepared to present in class**.

The **minimum** expectation is a **detailed wireframe prototype**. However, the final prototype required for user testing must be programmed to be interactive. DO NOT DELAY too long in starting development.

Submit:

- **Prototype screens**. For each screen in your prototype, have a screen ID, title, and a short description as to what system features the user and computer interaction represents.
- **Design rationale** – what design guidelines and principles were used?
- Use the **Interactive Design template**.
- **Team status report**.

Deliverable 6: Heuristic Evaluations (Done in Class)

Your prototype will be distributed to at least **two** of your **classmates**, who will do heuristic evaluations of it in class. Since your evaluators must be able to view and interact with your prototype, this will impact the extent or type of revisions that you implement in your system. If you have a programmed prototype, be sure your system works/is viewable in the labs/team rooms. Given the time constraints, your system may not be completely refined (prioritize) for heuristic evaluation. At a minimum you need to be able to support the **five tasks** you identified in your **task analysis**. Prepare a HTA based description of each task on an index card ahead of the evaluation.

Each evaluator should be paired with an observer from the team who will fill out the "Heuristic Testing Worksheet". After the evaluations are complete, consolidate the individual observations into one problem list. As a team assign each of these problems a severity rating (cosmetic, minor, major, catastrophic), and brainstorm possible solutions for it. Plan system revisions to correct as many of the problems found as possible (in priority order) in time to begin user testing.

Submit an evaluation report that includes the following information to the "Heuristic Evaluation" dropbox.

- The two original heuristic testing worksheets
- The consolidated problem list with severity ratings
- Summary of the teams problem analysis and plan forward

The deliverable will be graded based on the completeness of the material submitted. Note: if you aren't in class for this, then you don't get credit.

Team status report.

Deliverable 7: Test Plan and High Fidelity Prototype

You need to be done with the development of your system's prototype to at least "beta" quality in preparation for user testing. That means all features to be tested and the associated controls and aesthetics must be done. You will likely need to have simulated the back end of at least parts of your system, depending on its complexity. The completed prototype "beta" release should include an executable file or URL, and a README file. A good README file contains information on system installation, operation, and unsupported features and bugs. A good README file contributes to a successful UX.

To accomplish user testing you first need to develop your Usability Evaluation plan and materials. As part of your plan, you will test your system with the same five tasks used in prior deliverables. Use the provided **test plan template**.

Find at least **four more** representative users for a total of eight users who will test your system. None of your users should be enrolled in SWEN444. You should strive to have all of your users be members of your target population, but that is likely not possible. **All** participants need to be volunteers and must sign the **Informed Consent Form**.

The evaluation session must be scripted, so that you do and say the same things for each user (including describing what the purpose of the system is). I highly suggest doing a pilot test with someone else in this class to work out any issues that could jeopardize your results. For each user your script should portray when you provide the tasks (one at a time), and how you observe and take notes. One member of your group should be the facilitator of the test, one should be timing tasks, and one should be taking notes.

After collecting all of the user data, analyze this information to derive any usability problems found by your user tests. These problems should be put into a **list**. I suggest you consolidate this list with the problems identified during heuristic evaluation. Assign each problem a severity rating as above (cosmetic, minor, major, catastrophic), and brainstorm possible solutions for the problems. Then fix your implementation to solve as many problems as you can in the time available, giving priority to severe problems. You may not be able to get to all of them, but you should at least address the severe ones wherever possible. **It is important that you convey this analysis and what you did with the results in your presentation (Deliverable 9).**

Team status report.

Deliverable 8: Test Data and Analysis

Submit the **signed consent forms**. They may be handed in on paper or as scanned electronic documents (preferred). Submit the collected qualitative and quantitative **raw test data and the data analysis performed**. Summarize the test data analysis:

- Outliers
- Quantitative and qualitative data correlation
- Testing goals met

- Problems identified with severity ratings in a consolidated problem list, and usability solutions to address at least critical and major problems.

Team status report.

Deliverable 9: Presentation and Final Product with Updates

At the end of the semester, your team will give a 15 minute **presentation** of your project. Your talk will need to discuss the following:

1. Summarize the **system concept**.
2. Summarize the **interactive design requirements** in reference to the primary **work roles**. Describe the **usability requirements**.
3. Discuss how your **design evolved** through conceptual, intermediate, and detailed designs. Show sample screen shots. Explain your design rationale as the design evolved. What metaphors, affordances, and design principles were employed?
4. **Demonstration**. Demonstrate your design and implementation via a live demo of your system, working through the five tasks. How were usability requirements met in the design?
5. **Evaluation and Reporting**. Discuss the major findings from your evaluations (cognitive walkthrough, heuristic evaluation, and user testing). Include a discussion of the testing data you collected and your analysis. Did you meet the usability requirements? Show traceability between problems identified and subsequent changes to the design.
6. **Reflection**. Reflect on what went well and what could be improved in the project, particularly relating to usability but also for general software engineering concerns.

Be sure to turn in your **slides** by the due date and NOT when you give your talk. There will be a short Q&A after the talk that is not counted towards the 15 minutes. There is a rubric for your reference in myCourses.

In addition, turn in the final release of your project (the running prototype) with a **README** file. A good README describes the installation procedure and runtime environment, lists unsupported features and bugs, and provides introductory user operation instructions.

Where possible, correct critical usability issues identified in the usability testing. It will be subjectively graded from several perspectives. Is this a "good" interface to support user goals and a positive UX? Are severe usability issues addressed? Is the prototype sufficiently complete to yield meaningful user testing results? Degree of difficulty and the state of completion versus the specified requirements are also factors.

Team status report.