Spring 2020 CS151 S4-S5 Team Project

Dr. Katarzyna Tarnowska

Contents

Format

Team formation

1. The team project is to be completed within the teams of (exactly) three members.
2. The teams can be cross-sectional, although members within one section are preferred.
3. Each team has to declare its members on Canvas discussion forum prior to start working.
4. The instructor will accept the team, assign the team ordering number (team identifier that will be used throughout the semester), and create Canvas page for the group project. It is not possible to change the group after the assignment of the ordering number.
5. The team should decide on one of four available subtopics (see section Scenario, subsection Topics) and declare its choice on Canvas. There will be exactly four teams working on each subtopic. First

Canvas-declared, first-taken rule. If the topic reaches the limit of four teams, the team must decide on a different subtopic.

Teamwork

1. You are encouraged to work in team, however given the current circumstance, it is advised against in-person collaboration or in-person meetings. Utilize online collaboration tools and communication or divide the work so that everyone works on a separate tasks/deliverable. Communicate often with your team members and be aware that some of them may be geographically distant.
2. Online collaboration ways:
   - Communication: e-mail, Slack, Skype, Canvas group page, Canvas messaging, Zoom
   - Online diagramming tools: Lucidchart (free with sjsu account)
   - Code version control: GitHub
3. Academic Integrity
   - No collaboration outside own group is allowed. Inter-group collaboration in any phase of the project is considered a violation of academic integrity.
   - Any code borrowed must be properly referenced in the comments. Otherwise it will be considered plagiarism.
   - Any violation to the above rules will result in 0 points for the entire project for the entire team and reporting the case to the Office of Conduct.

GitHub Repository

1. Each student needs to have a GitHub account, preferably a student account. You can update your GitHub account by updating your email address to sjsu.edu
2. Each team needs to set -up a repository on GitHub using the following naming scheme S20-CS151-<t>-Team<n>, where t – your topic number – from 1-4, n – is your team number as assigned by the instructor on Canvas.
3. Each team member has to be added to the repository as a collaborator.
4. To count your contribution, you must commit with one Git account only which is a recognized GitHub user (otherwise GitHub will not count your commits and therefore your contribution will not count towards the grade).
5. The repository has to be set a private (means not publicly visible)
6. You need to add instructor (`tarnowska`), TA (`JVP15` if your team number is odd number or `susmitag` if your team number is an even number), as well as tutor if you wish a tutor support for the project (you need to ask tutor for his GitHub account).
7. Note: if private repositories are set by a non-student account the limit of collaborators is 4.

Contribution

1. Each team member has to contribute in the equal magnitude of effort to the group work. There is one submission per team, but grades will be assigned individually based on contribution.
2. The contribution of a team member is measured by:
   - Average of the percentage contribution assessment given by the other two members (confidentially to the instructor/TA) for the phase submission.
   - Net code lines added/removed on the GitHub repository by an account held by a member. Automatic code generation (such as Javadoc) are excluded from that count. DO NOT COMMIT automatically generated code, such as JavaDoc on GitHub.

Project Scenario

Each team will be working on implementing a clinical information system. The system is supposed to support transactions in a clinic that treats hearing disorders using tinnitus retraining therapy [1]. A basic transaction is registering a new patient. There should be fields to enter patient's: identification number in the clinic (ordering number), the date patient added to the system, patients first and last name, date of birth, gender, phone number, address (fields for street address 1, street address 2, city, state, zip, country), social security number, insurance number. Additional information about demographics, such as occupation, work status, educational degree of a patient can be added into the system. Finally, the system will keep track about etiology and onset of both tinnitus and hyperacusis, and any additional discretionary notes entered by medical personnel. There should be options to save the patient's data, cancel the add transaction, and add a new visit to the current patient. The patients added to the system can be viewed (i.e. in a table format) and edited.

Another important basic functionality is adding a new visit for the patient registered in the system. The system should also allow to view and edit existing visits. There are initial visits to the clinic and follow-up visits. For each visit we need to register the patient, the date, the visit sequence number (0-initial visit, 1-first visit, 2- second, etc.). At initial visit an initial interview is performed. A structured questionnaire called Tinnitus/Hyperacusis Initial Interview Form [2] is used to gauge – diagnose and quantify the patient's hearing problem. The audiology is performed mainly at initial visit but may also be performed as a part of subsequent visits. Medical history is gathered from patient and all the current medication that patient is taking are entered into the system. Based on all these information the physician assigns a category to a patient: 0-tinnitus present but no impact on life; 1- tinnitus present and has high impact on life; 2- hearing problem present and relevant; 3- hyperacusis (decreased sound tolerance) is a major problem; 4-prolonged tinnitus exacerbation. After consulting with the patient, the physician determines the treatment protocol (0-4). Tinnitus Handicap Inventory (THI) and Tinnitus Functional Index are used as standard ways to assess the impact of tinnitus and measure the treatment progress at subsequent visits. At each visit we register the treatment progress and treatment methods applied, including sound therapy, real-ear measurements, and counseling. Additionally, the system should allow to schedule the date of the next visit.

Subtopics

Besides the basic functionality (patient and visit management) each team must decide on one additional component among four possible to choose. For each topic, there should be at least one graphical drawing related to that topic, that support understanding medical information, such as a graph depicting progress of treatment or a graph simulating audiogram result. The invention and creativity of the team matters here.

Topic 1: Registering patient interview using structured questionnaires (Tinnitus and Hyperacusis Initial and Follow-Up Interview Forms) and physician's diagnosis at initial visit (the diagnosis is one of the five categories 0-4, as defined by TRT). The system should allow to enter all information and questions as described in the questionnaires [2-3].

Topic 2: Registering treatment progress using Tinnitus Handicap Inventory [4-5] and Tinnitus Functional Index [6-7]. The system should allow to enter answers for al the 25 questions in each questionnaire. The system should also calculate the total score of each questionnaire and score per category. The system should determine and display the severity based on the mapping of the total score to a handicap category [4-7].

Topic 3: Performing audiological evaluation and registering medical history including pharmacology. The audiological evaluation includes: pure-tone audiogram for the left and right ear in all frequencies (from 0.25 kHz to 12kHz), loudness discomfort levels (LDL) for the left and right ear in frequencies 0.5 kHz to 12 kHz, tinnitus pitch match and match type, thresholds of hearing, minimal masking levels for the left and right ear. Physician should be able to enter any additional discretionary comments regarding audiology. The system should allow to record any current medication the patient is taking, including the medicament's name, generic, dose, duration, chemical category, action, application, usual dose, and whether the medication induced tinnitus as a side effect.

Topic 4: Registering treatments applied at the visit, including details of sound therapy, real-ear measurements, and counseling. Sound therapy is administered by the use of sound instruments. There are different categories, models, and types. Below is a sample hierarchy of different categories of sound instruments:

- Sound Generators (SG)
    - Types:
        - Viennatone (V), models: AmTi, BTE
        - GH-Hard (GHH), models: ITE, ST, TRI-COE
        - GH-Soft (GHS), models: ST, TRI-COE
- Hearing Aid (HA) - types:
    - HA: models: AIR-BTE, BTE, BTE P, CIC, CROS, Interton, ITC,ITE, Jazz-COE, Oticon, Phonak, Vivatone
    - CO: models: TCI-C
- Combined Instruments (CO) - types: CO, models: BTE, TCI-C, TCI-COE

Real-ear measurements are administered to help fit the instruments to patients. The following numerical measurements are taken as a part of REM:

- Right ear - Freq RE, Th R SPL, Mix R SPL, Mix R SL, Tol R SPL, Tol R SL, Max R SPL, Max R SL,
- Left ear - Freq LE, Th L SPL, Mix L SPL, Mix L SL, Tol L SPL, Tol L SL, Max L SPL, Max L SL

Counseling is education and teaching tailored to the specific need of the patient. The system needs to register notes related to counseling provided at a visit.

Literature

[1] Jastreboff PJ. Tinnitus Retraining Therapy. In: Møller AR, Langguth B, DeRidder D, Kleinjung T, editors. Textbook of Tinnitus, Springer; 2011. Chapter 73.

[2] Jastreboff MM, Jastreboff PJ. Questionnaires for assessment of the patients and treatment outcome, Sixth International Tinnitus Seminar, 1999.

[3] Henry JA, Jastreboff MM, Jastreboff PJ, Schechter MA, Fausti SA. Guide to conducting tinnitus retraining therapy initial and follow-up interviews. The Journal of Rehabilitation Research and Development. 2003;40(2):159.

[4] Newman CW, Wharton JA, Jacobson GP. Retest stability of the tinnitus handicap questionnaire. Ann Otol Rhinol Laryngol 1995; 104: 718-23

[5] Download the THI here: https://www.ata.org/sites/default/files/Tinnitus_Handicap_Inventory.pdf

[6] American Academy of Audiology. Tinnitus Functional Index. Retrieved from https://www.audiology.org/news/tinnitus-functional-index

[7] Download the TFI here: http://download.lww.com/wolterskluwer_vitalstream_com/PermaLink/EANDH/A/EANDH_2011_09_27_HENRY_200593_SDC15.pdf

[8] Tarnowska KA, Ras ZW, Jastreboff PJ. Decision Support System for Diagnosis and Treatment of Hearing Disorders. The case of Tinnitus. International: Springer; 2017.160 p. (free online copy available from Atkins library).

Milestones

Analysis & Design Phase: due 4/15 11:59 PM, 100 points

Deliverable

1. System analysis & design: use case analysis, complete UML documentation, Javadoc design documentation for the system (no implementation), and GitHub repository set up.
   1.1. Use case analysis: a complete list of use cases and their sequences that depict interaction between the user and the system. Use concrete user events and concrete components you propose in UI Design (see point 2), such as "clicks the Save button".
   1.2. UML use case diagram that depicts all use cases identified in the scenario and the chosen subtopic.
   1.3. A complete UML class diagram that follows Model-View-Controller design pattern (a complete separation of data model from the user interface). The class diagram must be detailed in GUI classes design up to a frame and panel level.
   1.4. Two UML sequence diagrams for the chosen, representative use cases.
   1.5. A UML state diagram, that depicts navigation within your program.
2. User interface (UI) Design: mock interface design for all screens and all components within a screen
   2.1. Include wireframe design screenshots for each screen with a detail up to a component and event associated with the component
   2.2. Justify your design choices (such a layout, panels, components) with the text description. Justify your choice based on the following criteria: intuitiveness, navigability, graphics, usability, user-friendliness. See "10 Usability Heuristics for User Interface Design" by Jacob Nielsen.
3. Link to a GitHub repository and a GitHub account username for each team member.
4. Design documentation committed on GitHub (committed comments without implementation) and generated JavaDoc attached to the submission.

Canvas submission

There should be one report with all sections addressed in the Deliverable section, including diagrams embedded into the report. JavaDoc should be separate – compress generated html file into one file. Then compress the report and compressed JavaDoc into one file and submit as one file on Canvas. Since this is set as group project assignment, only one submission per team is required.

Tools

1. Report, Use Case analysis: Google docs or similar tool allowing for online collaboration.
2. UML: Lucidchart (online diagramming tool, allows for online collaboration, free with sjsu account).

3. Mock wireframe design: a tool of your choice, examples: Mockflow (web-based), MS Visio.
4. Code & Documentation: GitHub and Git (see tutorials uploaded on Canvas).

Grading criteria

Criteria	**Points**
System Analysis & Design	**0-50**
Use case analysis & use case diagram • If problem and scenario requirements properly understood • If complete set of use cases identified • If use cases described in detail and correctly • If proper UML notation used	0-10
UML class diagram • If a complete set of classes identified • If proper relationship between classes identified • If classes identified in detail • If MVC incorporated into the design • If UML notation complete (including methods) and correct	0-10
MVC pattern • If MVC design paradigm followed • If MVC properly incorporated into the design	0-10
UML sequence diagram • If complete and complex enough • If correctly represent the scenario • If correct UML notation used and consistent with the class diagram	0-10
UML state diagram • If correctly and completely represents the states of the application • If uses correct UML notation and consistent with the class diagram	0-10
User Interface (UI) Design	**0-30**
UI design • If UI Design is detailed and well thought-out • If mock design used and screenshots provided • If UI is well designed following specified criteria and usability heuristics • If the choice of UI layouts, containers and components is adequate and correct • If uses a variety of components adequate for functionality • If the design choices are well justified and described	0-30
GitHub and Design documentation	**0-20**
GitHub and Git • If GitHub repo set up with all the requirements specified in this document • If each team member was able to commit with a recognized GitHub user • If each team members contribution described in detail	0-10
Design documentation & Javadoc • If design documentation used and committed on GitHub • If Javadoc correctly generated and complete	0-10
Total Note: the individual grade will be adjusted based on the contribution.	0-100

Implementation Phase: due 5/11, 200 points

Implement a working and interactive application that follows MVC design paradigm and meets all requirements specified by written scenario and any additional clarifications provided by the instructor.

Deliverable

1. Source code (src)
2. Documentation (doc) generated using Javadoc utility
3. Final Report (11pt, Arial, single-spaced):
   - 3.1. Problem statement:
     - 3.1.1. Tinnitus as a medical problem with reference to the literature (1/2 page)
     - 3.1.2. Tinnitus Retraining Therapy description with focus on the diagnosis/treatment protocol (1/2 page)
     - 3.1.3. Clinical Information System – the goals of the system and TRT transactions supported by the system (3/4 page)
   - 3.2. Design: include all the results from Phase 1 (diagrams and descriptions) modified with the feedback received or changes to the design throughout implementation.
   - 3.3. Implementation:
     - 3.3.1. Screenshots of all implemented screens (frames)
     - 3.3.2. Description of implementation, including application of MVC pattern, interesting parts of codes related to the topic learned in the course, such as using lambda expressions, inheritance hierarchies, interfaces, abstract classes, enumerations, any additional design patterns. Description of any special and interesting Java Swing components.
     - 3.3.3. Description of testing methods (i.e. test cases, Junit tests)
   - 3.4. Short deployment instructions (AKA README).

Canvas submission

If you are using Eclipse, compress the entire Eclipse project and together with the report into one file. If you are not using Eclipse compress your source code (src), documentation (doc) and report into one file. One Canvas submission per team.

Tools

1. IDE: preferred Eclipse, but other allowable
2. Java: version at least 8 (preferably all team members work with the same Java version)
3. Git: you need to have Git configured in your IDE (for Eclipse- see tutorials).

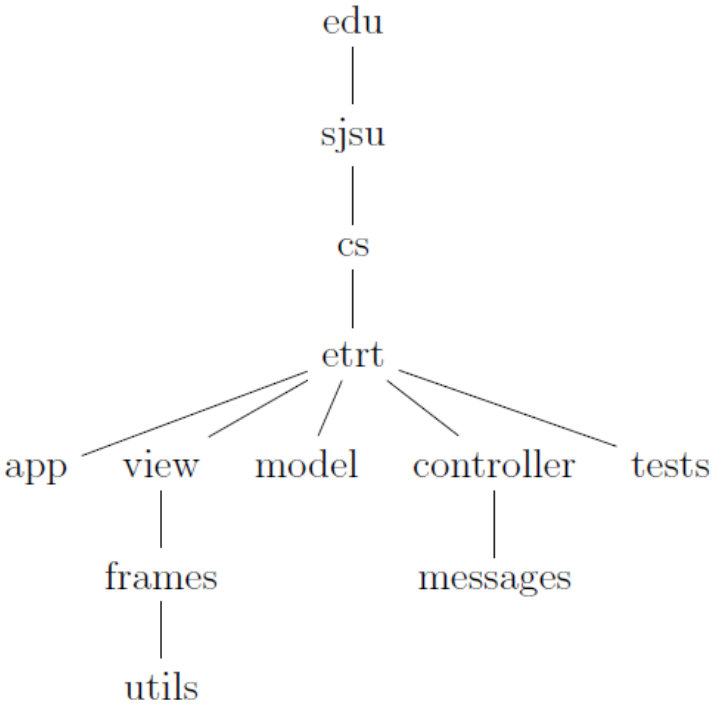
Grading criteria

Criteria	**Points**
MVC Implementation	**0-80**
Model • Appropriate classes for the data Model identified • Object oriented design principles followed: encapsulation • If additional design patterns identified and implemented when appropriate • Uses class hierarchies (inheritance) and interfaces when appropriate • Uses appropriate data structures and implements correct and efficient algorithms • If separated from the GUI (View) – cannot reference any GUI components	0-30

View • Implements 'user-friendly' I/O interface with GUI components • Uses a variety of Swing/AWT components • Incorporates graphics (i.e. chart, a drawing) into UI • Proper user interaction and action listeners implemented • If separated from data Model – cannot reference Model classes directly	0-30
Controller • If communication between Model and View is entirely through Controller • If Controller implemented efficiently • If application is responsive to the user input and events	0-20
Functionality implementation • If properly understood and implemented • If functionality working correctly, if complete and tested	**0-50**
Basic • Adding a new patient • View/Modify patients • Adding a Visit (with all the TRT components, but leave the ones that are irrelevant to your subtopic unimplemented) • View/Modify visits	0-20
Additional functionality specified in the subtopics (i.e. questionnaires, audiology, etc.) Note: projects without additional functionality in subtopics will not be accepted	0-30
Testing • If testing thorough, robust, and extensive • If uses Junit framework	**0-10**
Code Documentation • If code properly documented • If proper conventions followed • If documentation thorough • If Javadoc generated	**0-20**
Report	**0-30**
Problem statement • If complete, well-researched, and well-written • If references literature	0-10
Design • If complete and consistent with implementation • If design efficient, object-oriented and using MVC pattern • If feedback from Phase 1 incorporated	0-10
Implementation description and instructions • If complete and well-written • If implementation choice well-justified • The instructor/TA able to deploy the system based on instructions provided	0-10
Teamwork and Code Version Control	**0-10**
• If evidence of good teamwork • If proper use of Git and GitHub • If regular commits evident • If contributions of all team members similar in the order of magnitude	
Total (the individual grade for the project will be adjusted based on the individual contributions).	**0-200**

Model-View-Controller Pattern

The high-level architecture (package structure) of your system should be as follows (eTRT is a sample name of the application, you may use a different name for the application).



- **app** — Initializes the eTRT desktop application. Does not include any business logic or GUI implementation.
- **model** — Contains all business logic of eTRT such as Patients, Visits, Audiology, and Counseling.
- **view** — Contains all of the GUI code, made up of three sub-packages:
  - **root** — Contains the monolithic View class which controls all GUI actions.
  - **frames** — Contains the code for all eTRT GUI frames.
  - **frames.utils** — Contains static methods for shared use by various eTRT frames.
- **controller** — Manages communication between the **View** and **Model** package
- **tests** — Contains *JUnit* unit tests