CS 495 Capstone

“Where Would You Sit?”

Planning Document

Software Requirements Document (SRD)

Software Name:		Revision #:	1
Author:	David Vaughn, Daniel Klein, Brandon Laird, Kelly Galuska		
Date:	02/03/2015		
Revision History:		Date:	
Revision #1 completed		02/12/2015	

Table of Contents

1.0 Introduction

Product Description:

WWYS (Where Would You Sit?) is being developed at the request of Dr. Jason Scofield for his research into how adults and children use space differently. The product will feature a way for an experiment subject to select a chair in a room. It must also include a way for an experimenter to customize room size, chair numbers, and survey questions and sync this data to a database from which it can be accessed for analysis.

2.0 Team Members

List the names, titles, and roles of the project team members.

- **David Vaughn,** role to be determined.
- **Daniel Klein,** role to be determined.
- **Brandon Laird,** role to be determined.
- **Kelly Galuska,** role to be determined.

3.0 Assumptions, Constraints, Schedule and Design

3.1 Assumptions

- The hardware to support the application and run the experiments is already in place; this includes laptops, tablets, and a server.
- Android will be the operating system upon which the application will run, and a near-identical web application will be developed concurrently.

3.2 Constraints

- **Schedule**: application development and testing must be complete by early April 2015.
- **Budget**: budget for the development of the application is very low.

- **Security**: both the database for experiment data and the application itself will need to be secure.
- **Languages**: the application will be developed using HTML, CSS, and Javascript.

3.3 Schedule

- **Tasks**: Schedule of Tasks for Developing each Deliverable Item. Additional schedule items may be needed to manage the project as work progresses.
    - **Task:** 2/3/2015, begin work on Requirements Documentation (SRD) and accompanying UML Diagrams
    - **Task:** 2/13/2015, begin development of project website
    - **Task:** 2/18/2015, begin work on Design Documentation and accompanying presentation
- **Milestones and Deliverables**: Schedule with dates of major milestones and deliverables that result from completion of the project tasks.
    - **Deliverable:** 2/12/2015, requirements documentation and UML, along with accompanying presentation
    - **Deliverable:** 2/17/2015, initial version of project website
    - **Deliverable:** 3/3/2015, design documentation, detailed UML diagrams, and accompanying presentation
    - **Milestone:** 3/4/2015, begin implementation of application
    - **Deliverable:** 4/11/2015, project progress demo
    - **Deliverable:** 4/23/2015, full source code, video, website, and acceptance testing

4.0 General System Description

4.1 System Context

The project will consist of an Android mobile application and a virtually identical web application, developed from the same base of source code. This document will refer to both interchangeably, as they are derived from the same source code base. Each will be self-contained and will interact with a back-end server that will handle data processing and data transfer to final persistence location.

4.2 System Environments and Modes

In the interest of simplicity and low cost, development of the system will require only two environments. Development and initial testing will be performed on the developers' local development machines, emulated Android devices, and physical Android test devices. Integration testing and production will occur on the environment to which the application is deployed; this will consist of the application server along with Android devices and web browsers designated for system testing. Because of the infrequent and light load this system expects to support, no complex back-up or redundancy scheme is necessary, and a simple single-server architecture will suffice.

4.3 User Characteristics

This system will have two types of users: experimenter and experiment subject, with differing use cases. Thus, the application will be divided into two sections. One section will consist of experimenter tasks; these tasks include configuring the parameters of an experiment. The other section will consist of tasks performed by the experimental subject; this section will consist essentially of only the choosing of a seat in the virtual room. Both of these sections will require an intuitive and easy-to-use graphical user interface, as neither the experimenter or experiment subject is assumed to have a technical background, and it is likely that many of the experiment subjects will be children.

4.4 Operational Scenarios

The system has essentially a single operational scenario: in this scenario, the experimenter configures an experiment setup and locks the configuration; the experimenter then makes the application available to one or more experiment subjects, allowing each of them to select a seat in the virtual room depicted on the device display.

4.5 Standards, Procedures, and Processes Used in this Project

The project developers will ensure that all source code adheres to idiomatic standards of the implementation language(s). All code will be readable, with comments where necessary, and all functions and methods will have appropriate docstrings. As discussed elsewhere in this document, all check-ins will be reviewed by at least one developer other than that who made the check-in, and all requested changes and fixes from review will be incorporated in a timely matter.

5.0 Functional Requirements

5.1 An experimenter shall be able to configure an experimental setting, including virtual room dimensions, number of chairs, configuration of chairs, and number and location of room entrances/exits.

5.2 An experimenter, after specifying the room dimensions (number of chairs), shall be able to remove chairs from the room or mark chairs as “occupied”.

5.3 An experimenter shall be able to configure the questions to be asked of the experiment subject prior to chair selection.

5.4 An experimenter shall be able to lock the experiment configuration and administer the experiment to multiple subjects using the same configuration before unlocking and editing the configuration.

5.5 The system shall be able to present the configured questions to the experiment subject and receive and save his/her answers.

5.6 An experiment subject shall be able to input his/her answers to the pre-experiment questions

5.7 Upon answering the pre-experiment questions, an experiment subject should be able to navigate their representative avatar around the room in some way and ultimately select a chair in which they would sit.

5.8 The system shall be able to store all input data in its database and export all experiment data to an external location and format for use by the researchers for analysis.

6.0 Interface Requirements

There exist two primary user interfaces: that for the experimenter and that for the experiment subject. Both will be graphical, intuitive, and aesthetically pleasing. The user interface for the experimenter will allow the experimenter to enter and edit questions that will be asked of the experiment subject before the subject selects a seat in a room. The experimenter’s user interface will also let him/her configure the parameters of the virtual room from which the subject will select a chair. Interaction with this interface will be in the form of touch, gestures, and typing on the on-screen keyboard for the Android version of this application, and in the form of mouse clicks and keyboard entry for the web form of the application.

The user interface for the experiment subject will present the subject with the virtual room and prompt the subject to navigate his/her avatar to the seat s/he selects. The interface will provide some method for the subject to move his/her avatar around the room and ultimately to his/her seat of choice. Interaction with this interface will be in the form of touch, gestures, and typing on the on-screen keyboard for the Android version of this application, and in the form of mouse clicks and keyboard entry for the web form of the application.

The system will have one additional interface, that which connects the system’s database to the destination location where experiment data will be exported. This

will likely take the form of a simple web page that will allow a researcher to download experiment data.

7.0 Data Management

Data related to this system will be generated by each instance of an experiment. In other words, one piece of data will be generated and persisted each time an experiment subject chooses a seat in the virtual room, consisting of information about the experiment subject, the configuration of the virtual room, and the subject's seat selection.

This data will be stored on the application server using a database solution yet to be determined. Some method will be developed for exporting data from this database into a form that can be used by Dr. Scofield and his colleagues for analysis; this will likely be in CSV or Excel spreadsheet format.

8.0 Non-Functional / Operational Requirements

Describe the non-functional requirements; do not state how these requirements will be satisfied.

8.1 Security, Availability, Reliability, and Recoverability

8.1.1 Security -- With regards to security, due care will be taken to transmit and store experiment data securely. While we do not expect to store any sensitive or personally identifying information, security remains essential so as to protect the researchers' original data and maintain the integrity of their work. Furthermore, a PIN system will likely be implemented in order to ensure that only approved experimenters are permitted to configure and administer experiments.

8.1.2 Availability -- The application shall be available consistently during reasonable business hours for the duration of the experiment. Any maintenance on the application will be done after business hours.

8.1.3 Reliability -- The system shall operate correctly for any experiment instance, assuming reasonable use of the system by users.

8.1.4 Recoverability -- Experimental data will be backed up periodically and will be recoverable should the primary server fail. In the event of a server failure, the system shall be available again after no more than a few hours.

8.2 Maintenance and Support

Maintenance and support will be provided by the developers during the time frame of project development and test. Beyond this time frame scope, there is no guarantee of maintenance or support from the original developers. It is likely that the original developers will be able to provide periodic service for a limited time, but ultimate responsibility for the maintenance and support of this system in the long term will fall to the researchers commissioning the system.

8.3 Performance, Capacity and Scalability

As volume of users is expected to be quite low -- at most a few experiment instances occurring at one time, large capacity and scalability are not major concerns for this project. The system shall be able to support a reasonable number of experiment instances simultaneously and to persist the resulting data from simultaneous experiments without issue.

8.4 Technical Reviews, Audits, and Walk-Through

All code commits will receive peer review by at least one other member of the team upon commission to the source code repository. Minor issues found by aforementioned review will be fixed as soon as possible with further commits, while more extensive code problems may require a full roll-back of the commit and reworking before re-committing.

9.0 Training

The developers will provide basic training for use of the system to Dr. Scofield and any key graduate students involved in his research. The developers will also provide Dr. Scofield with the basics of downloading experiment data from the database. Training of any further experimenters, such as students in any of Dr. Scofield's classes, will be the responsibility of Dr. Scofield or his graduate students.

10.0 SQA Requirements

10.1 Quality Plan

10.1.1 Change Control -- Proposed changes to the scope of the project will require discussion by the entire project team and approval by all members of the team before being integrated into requirements, design, or implementation.

10.1.2 Configuration Management -- In the event of multiple development builds of the application, each build will be identified by the

number of the latest commit pushed to the repository for the source used for the build.

10.1.3 Release Control -- In order to release a version of the application, the source code must past all unit tests, integration tests, and user tests. Unit tests will determine that the software works properly, and user testing by the developers will ensure that the application meets the requirements.

10.1.4 Testing Description -- The development team will work together to devise unit tests for each method and class in the project requiring testing. Before committing a code change, the developer must ensure that the code passes all existing unit tests. The development team will work together to devise a set of integration tests that exercise all use cases of the system, and all unit and integration tests must pass prior to a release of the application.

10.1.5 Defect Tracking -- The development team will use GitHub's defect tracking functionality to log and follow defects that exist in the source code. Any fix for a recorded defect will require independent testing and verification by at least one other member of the development team before the defect can officially be marked as fixed.

10.2 Test Plan

- **Description**: The If the developer's approach to testing is already documented in the Quality Plan, that description is referenced here. Otherwise, the processes to be used are described. The following Test Plan sections contain a project-specific description of testing for this project.

- **Testing Approach**: The developers will make test cases using graph to provide coverage of the methods and classes. The testing will include unit testing as well as other forms.

10.3 Testing Schedule:

Testing Includes tasks and major milestones. Milestone examples are start and end of module or system tests, system builds, test script creation, and regression testing. These dates are integrated into the master project schedule.

10.4 Documentation Plan

- **Planned Documentation:** Planned documentation includes use case diagrams, activity diagrams, readme, and integrated online help for the users. The use case and activity will provide other developers the

understanding on the original use of this program. A readme file will also be included and provide a small overview of how to install and use this application. The online documentation will provide users explicit instructions for how to use and install the program.

- **Documentation Schedule**: The document schedule is included within the original schedule

10.5 Delivery, Installation, and Acceptance

- **Installation**: Describes the planned method for installation: done by the user independently, done by customer company internal IT services, done by an external contractor. Specifies the handling of such items as data transfer from prior releases, and the presence of software elements from prior releases.

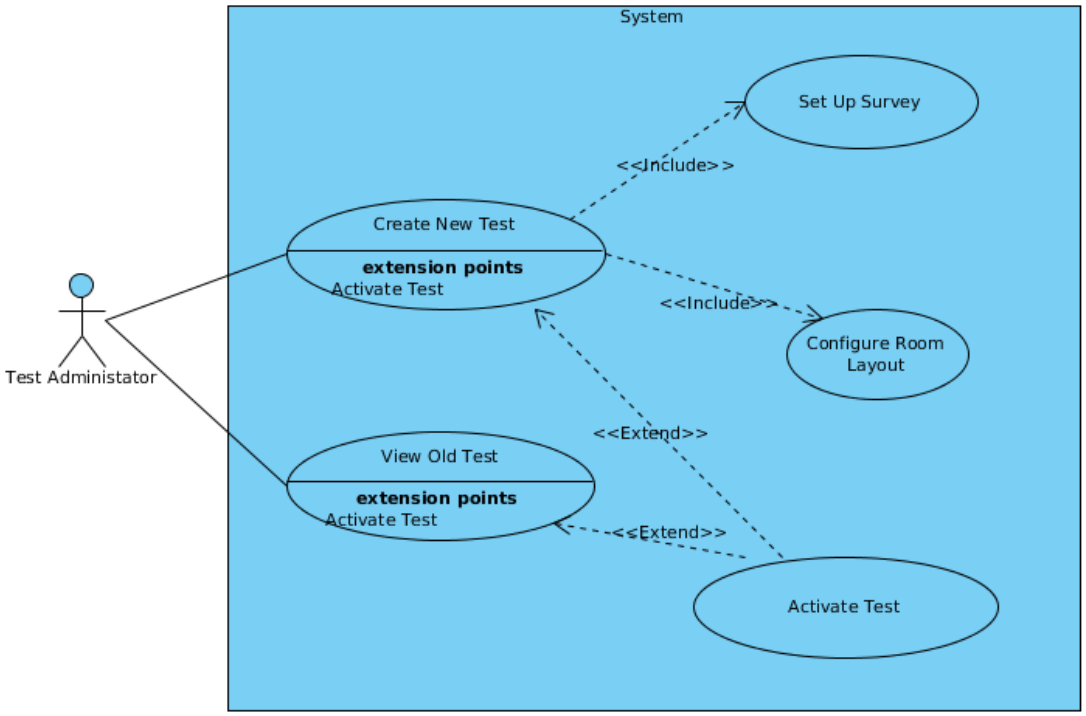
- **Usability**: With the Android operating system combined with the Bootstrap framework, the application will have an intuitive, easy-to-use interface, with straightforward controls and a simple-to-understand flow of use.

- **Acceptance**: Acceptance will be performed by Dr. Scofield and his graduate students.

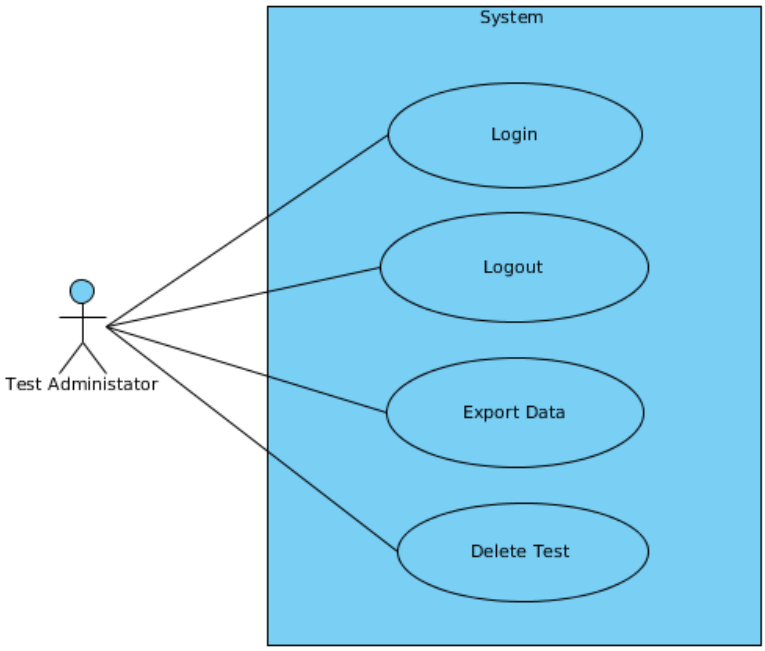
11.0 Analysis Diagrams

11.1 Use Case Diagrams

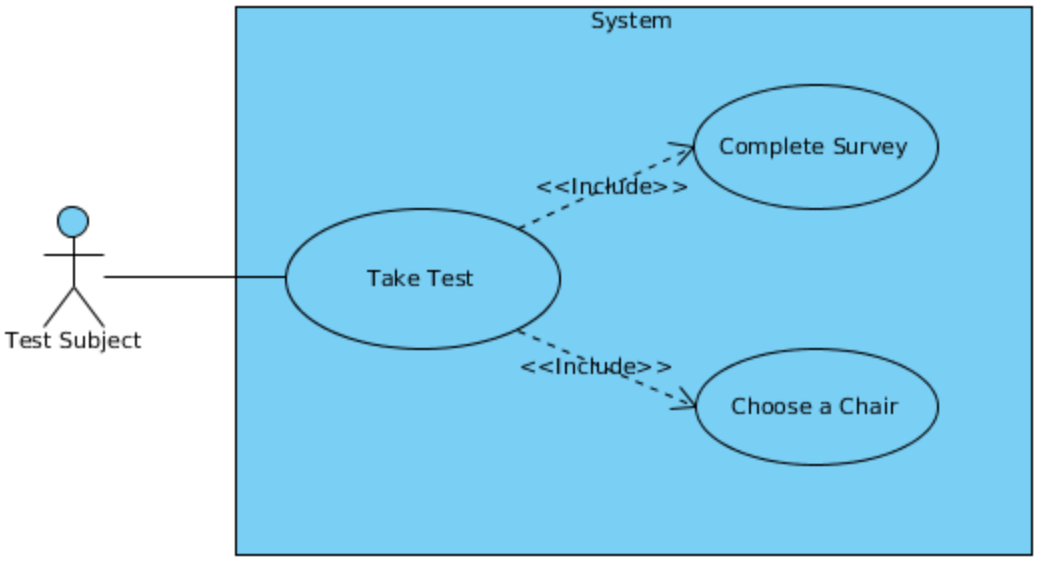
- The test administrator can participate in the Create New Test and the View Old Test use cases. During the Create New Test use case, the admin also uses the Set Up Survey and Configure Room Layout use cases as part of the test creation process. At some point in both the Create New Test and View Old Test use cases, the admin has the option to perform the Activate Test use case, but this is not mandatory.



- The test administrator can also use the Login, Logout, Export Data, and Delete Test use cases.

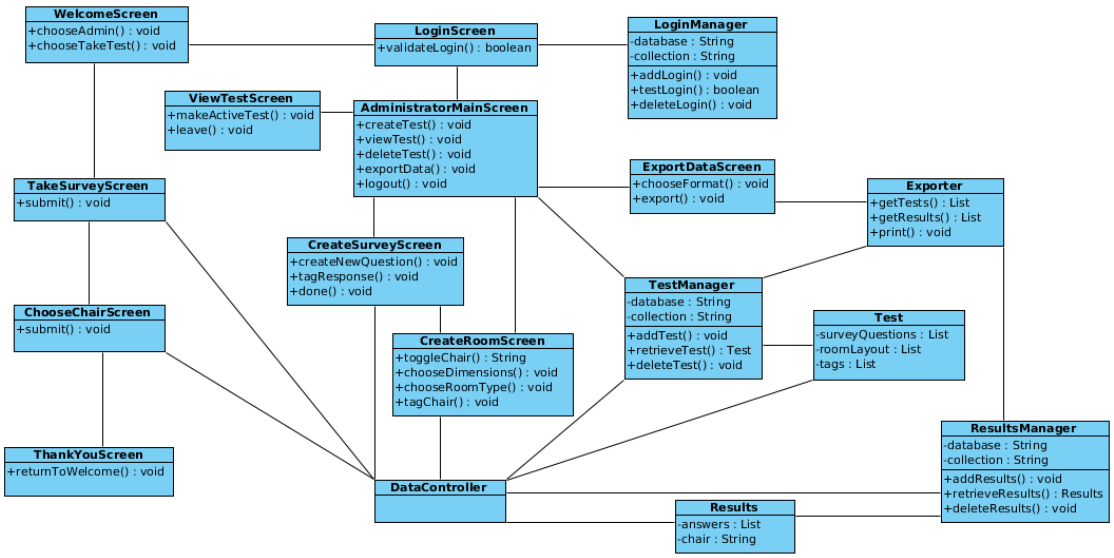


- The test subject only has the Take Test use case. While taking a test, the Test Subject also uses the Complete Survey and the Choose a Chair use cases.



11.2 Class Diagram

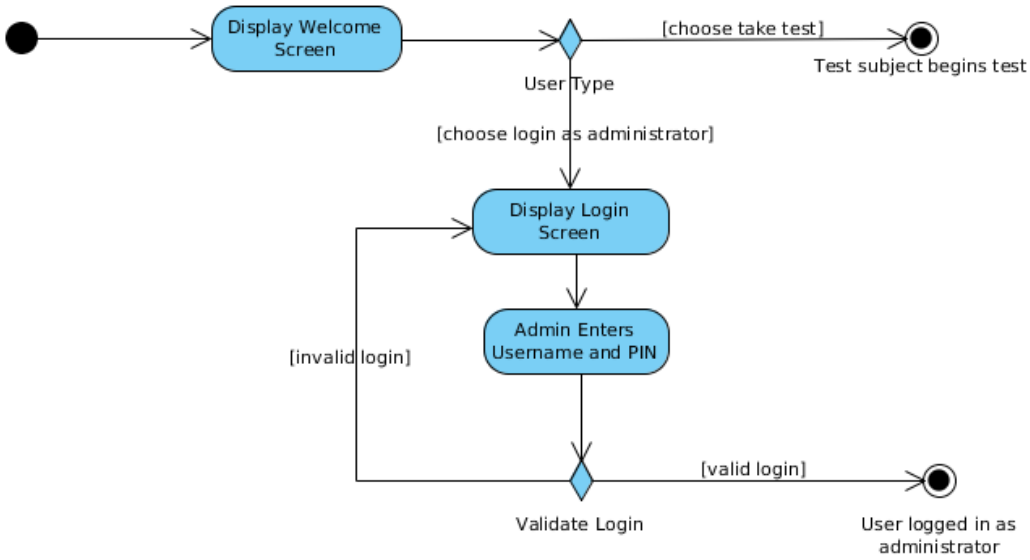
- **Description:** Each class with “Screen” in the name represents a user interface that will be implemented as an html page. The WelcomeScreen class is the first screen that is displayed when the program starts up. When taking a test, the user will be taken through the TakeSurveyScreen, the ChooseChairScreen, and the ThankYouScreen in that order. The LoginScreen is where test administrators can login to the system. The AdministratorMainScreen displays all the options available to an administrator. The CreateSurveyScreen and CreateRoomScreen are the interfaces that the admin can use to construct a new research test. The admin can look at existing test on the ViewTestScreen and can specify what data to export and what format to export it in on the ExportDataScreen. The Manager classes are interfaces for the system to interact with the database. Each Manager class is used to access a different collection within the database. The LoginManager handles login credentials, the TestManager handles parameters for tests that have been created, and the ResultsManager handles research data collected from the test. The Test and Results classes are abstract data types used to represent a set of test parameters and a set of collected research data, respectively. The DataController will be a means of allowing the user interfaces to effectively communicate with the database managers.



11.3 Activity Diagrams

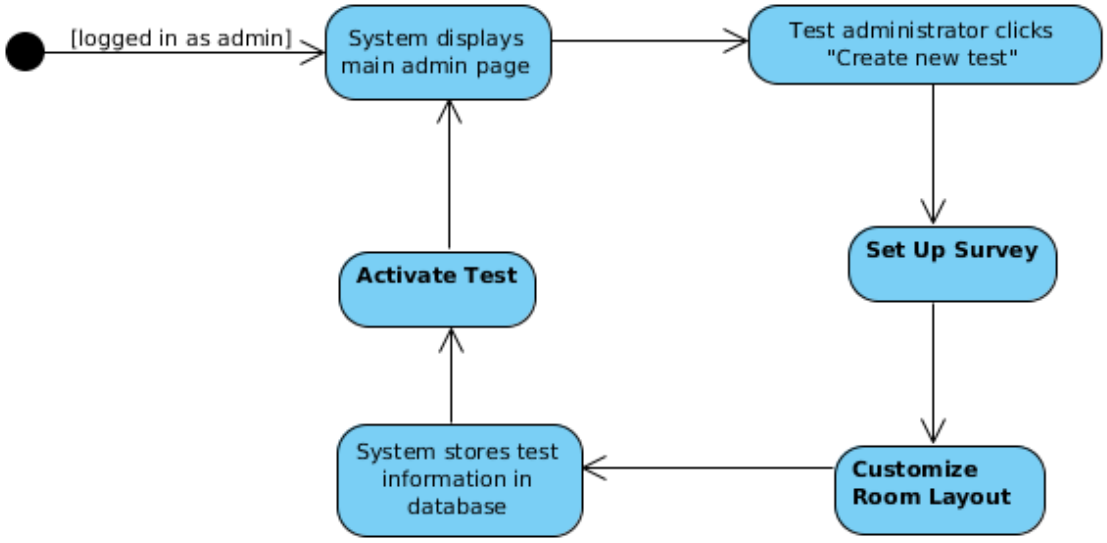
- **Use Case Name:** Login
- **Summary:** Admin logs in to the system with a username and password
- **Actor:** Test Administrator
- **Precondition:** None

- **Description:**
    1. System displays the welcome screen
    2. Admin chooses the login option
    3. System displays the login screen
    4. Admin enters his/her username and password
    5. If the system recognizes the login credentials, the admin is given access to administrator tools.
- **Alternatives:**
    - If the system does not recognize the login credentials, the system denies access to the admin and stays on the login screen.

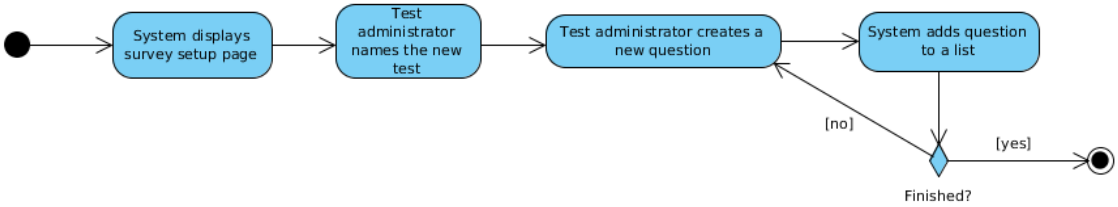


- **Activity Diagram:** Create New Test
- **Summary:** Shows how to make a new survey test
- **Precondition**: Administrator Login
- **Actor:** Test Administrator
- **Description:**
    1. The admin logs in then has the option to create a new
    2. The admin can then setup the survey questions
    3. Then setup the room layout
    4. Upon saving it stores the layout and questions are stored in a database.
    5. The admin is then allowed to activate the test.

6. It goes back to the systems main admin.

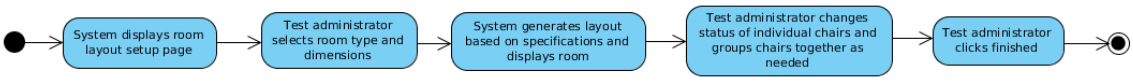


- **Use Case Name:** Set Up Survey
- **Summary:**
- **Actor:** Test administrator
- **Precondition:** Adminstrator sets up new test
- **Description:**
    1. System displays the survey setup page
    2. Admin names the test
    3. Admin creates a new question
    4. System adds the question to a list.
    5. If the admin is done adding questions, the program moves on.
- **Alternatives:**
    - The admin can choose to continue creating questions for as long as he/she likes.

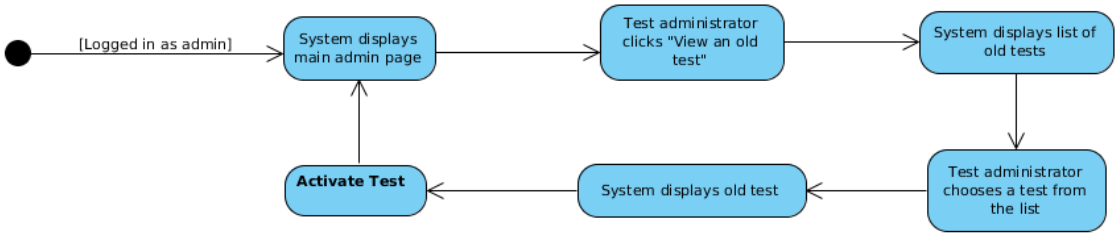


- **Use Case Name:** Configure Room Layout
- **Summary:** Allows admin to set up the room layout for a new test
- **Actor:** Test administrator
- **Precondition:** Administrator has just set up the survey for the new test
- **Description:**
    1. System displays room layout page.
    2. Admin selects room type and dimensions.

3. System generates and displays the layout according to the specifications.
4. Admin changes the status of individual chairs and groups chairs together as needed.
5. Admin clicks the finish button

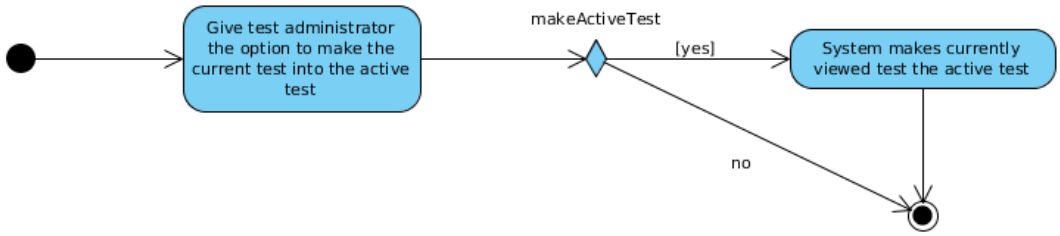


- **Use Case Name:** View Old Test
- **Summary:** Admin selects an existing test for viewing
- **Actor:** Test administrator
- **Precondition:** Admin has successfully logged into the system
- **Description:**
    1. System displays the main admin page.
    2. Admin chooses to “View an old test”.
    3. System displays a list of existing tests.
    4. Admin chooses a test from the list.
    5. System displays the chosen test and allows the admin to view the room layout and the survey questions from the test.
    6. Admin enters the Activate Test use case
    7. System returns to the main admin page.

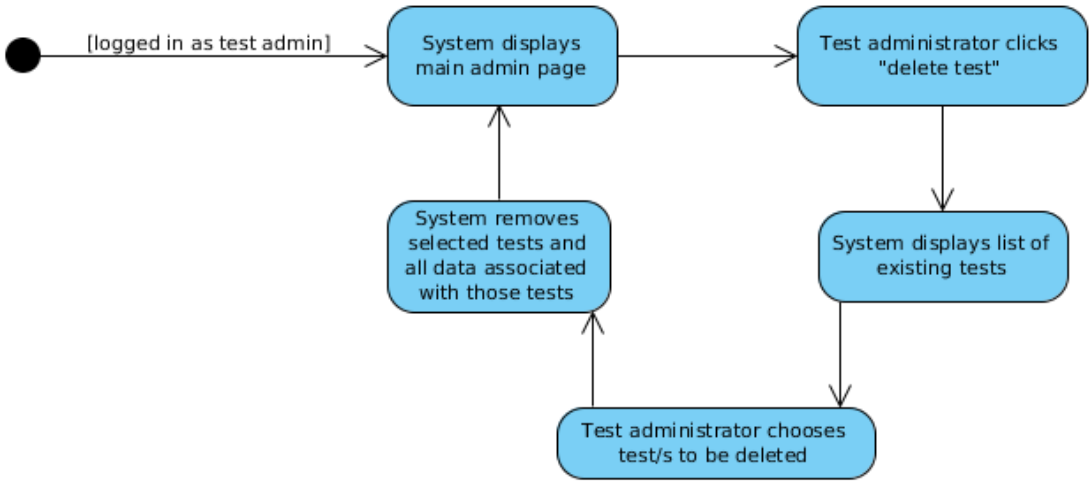


- **Use Case Name:** Activate Test
- **Summary:** Admin activates a test. The active test is the one that appears when subjects take a test
- **Actor:** Test administrator
- **Precondition:** Admin is either viewing an old test or has just finished creating a new test
- **Description:**
    1. System prompts the admin to choose whether to make the current test the active test.
    2. If the admin chooses yes, then the current test will be activated.

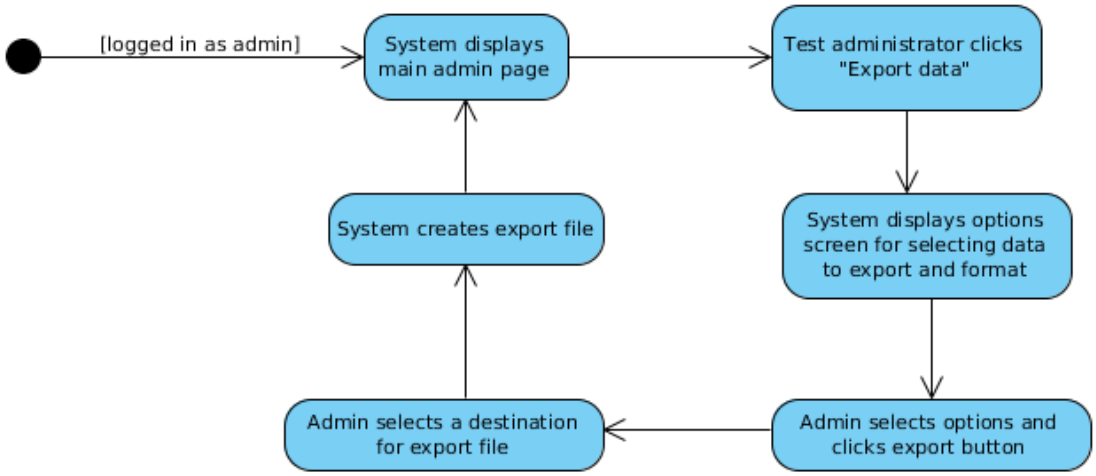
- **Alternatives:**
    - If the admin chooses no, nothing will happen (the test will not be activated)



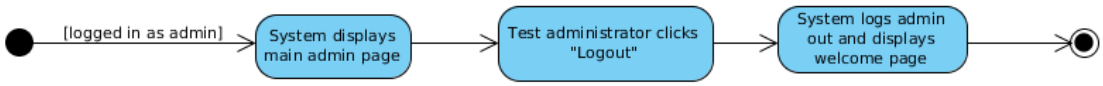
- **Use Case Name:** Delete Test
- **Summary:** Admin deletes a test and all data associated with it
- **Actor:** Test administrator
- **Precondition:** Admin has successfully logged into the system
- **Description:**
    1. System displays the main admin page.
    2. Admin chooses "delete test".
    3. System displays list of existing tests.
    4. Admin chooses one or more tests to be deleted.
    5. System removes selected tests and all data associated with those tests.
    6. System returns to the main admin page



- **Use Case Name:** Export Data
- **Summary:** Admin exports data from the database to a usable format
- **Actor:** Test administrator
- **Precondition:** Admin has successfully logged into the system
- **Description:**
    1. System displays the main admin page.
    2. Admin chooses “Export data”.
    3. System displays screen for selecting data to export and format.
    4. Admin selects options and clicks export button.
    5. Admin selects a destination for the export file.
    6. System creates the export file.
    7. System returns to the main admin page.



- **Use Case Name:** Logout
- **Summary:** The admin logs out of the system.
- **Actor:** Test administrator
- **Precondition:** The admin has successfully logged into the system.
- **Description:**
    1. System displays the main admin page.
    2. Admin chooses “Logout”.
    3. System logs admin out and displays the welcome page.



- **Use Case Name:** Take Test
- **Summary:** The test subject takes the active test

- **Actor:** Test subject
- **Precondition:** None
- **Description:**
    1. Test subject chooses “Take test” option on welcome screen.
    2. System checks the subject’s browser cookies to decide whether the subject has already taken the test.
    3. If the subject has not taken the test, the system displays the survey part of the test.
    4. The subject fills out the survey and chooses next to move on.
    5. The system displays the room layout page.
    6. The subject chooses a chair and submits the answer.
    7. The system displays a thank you page.
- **Alternatives:**
    - If the subject has already taken the test, the system informs the subject that he/she cannot retake the test.