EGR 401 – Capstone Design

Deliverable 5: Design Architecture, Detailed Design, and Detailed Budget

Team Name: Blueprints

Team Members: Jordan Ziegler

  Jeremy Fischer

  Justus Karenzi

  Sudi Nsengiyumva

  Dylan Shanahan

Client’s Name: Dr. Mark Gordon

Advisor’s Name: Professor Thomas Renck

Version: 2

Date: November 21, 2014

**5.1 Design Architecture**

**Level 0 Design**

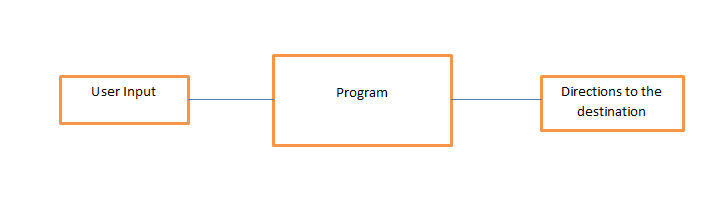


Figure 5.1.1 - Level 0 software architecture design

|  |  |
| --- | --- |
| *Module* | Program |
| *Input* | User Input |
| *Output* | Directions to the destination |
| *Functionality* | The augmented reality app will help students locate the rooms/offices. The user will input the room through a drop-down menu and the app will direct the user to his/her destination |

Table 5.1.1 - Level 0 Input/Output software Architecture

Our level 0 design includes all the modules that we will need for the project. The user will be able to choose a room from a drop down menu. Upon retrieving the room from the database, the app will help the user navigate to the corresponding room. Below in table 5.2 are all the modules with their respective inputs, outputs and functionalities.

|  |  |  |  |
| --- | --- | --- | --- |
| *Module* | *Inputs* | *outputs* | *Functionality* |
| **UI** | CSV | Drop down menu | The UI will provide the user with a friendly experience |
| **Drop Down Menu** | User Input | Directions | Easily select room |
| **Map Selection** | user input from drop down menu | map of building | This module will interact with the map of the building and retrieve the directions |
| **Image Processing** | Picture taken from device camera | Recognition of user location | Recognize the picture from camera to determine location of device |
| **Directions** | Current Location | User destination | Helps user get to their destination |

Table 5.1.2 Design Architecture Modules

**Level 1 Design**

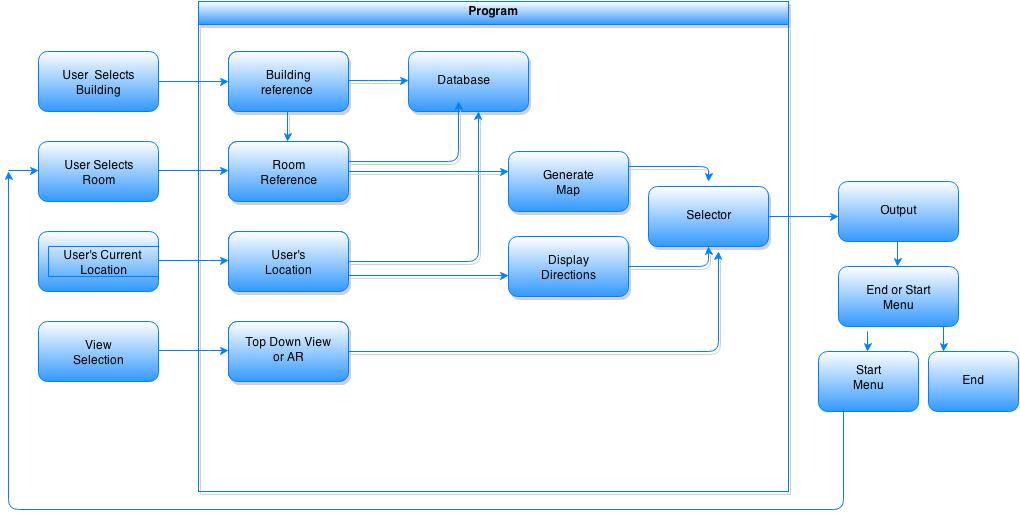
****

Figure 5.1.2 - Level 1 Design Flow chart

5.1.2: Level 1 Flow Chart Explanation

The above flowchart illustrates the way the user input will communicate with the program. Basically, the user will open the app and get to the home page where there will be an option to select a building that the user wants to navigate in. After the user select the building that he/she wants to go to, the program will go to the CSV database and pull up the room dropdown menu. After that, the user will select the room and input his/her current location. The program will go back to the Database to access information regarding directions and map. At that level the user will have an option to choose a Top down view map, Directions, or Augmented reality. When the user makes the decision of which view to display, the selector will select which view to display and output either the map, directions, or Augmented reality. After the user has reached the destination; they will be an option of ending navigation or going back to the main menu to look for another destination. If the user chooses the main menu, the program will go back to the home page otherwise the app will close.

**General Software Block Diagram and Explanation**

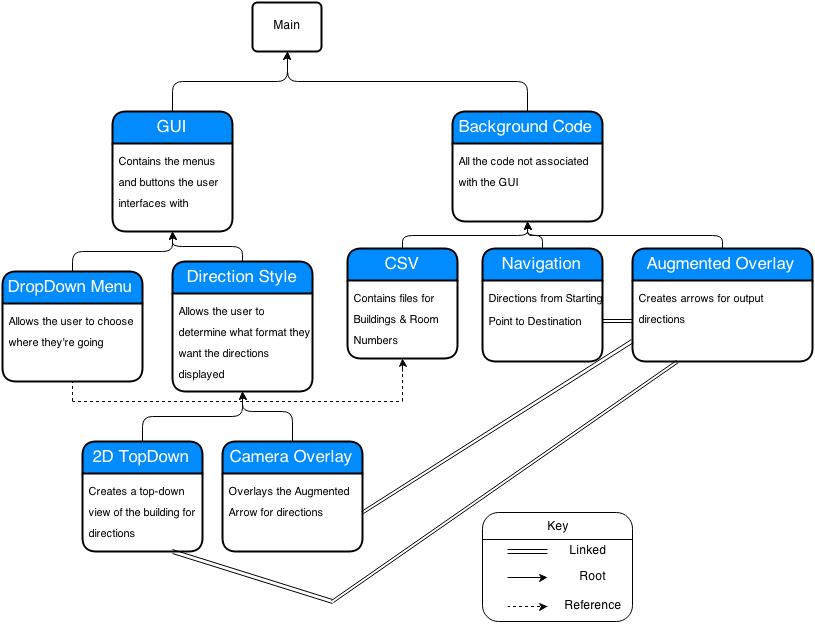


Figure 5.1.3 - General Software Block Diagram

The general flow diagram, shown above in Figure 5.1.3 shows the main parts of the program. The main program is divided into two sections, the GUI and background code. The GUI is the user interface and is subdivided into the different modules that the user interacts with. The user determines what building and room they are going to using the drop down menu. They also have the choice of output format, being able to choose between the augmented reality format and the more traditional 2D top down map picture. The background code is essentially all the code that happens behind the scenes, which the user does not interact with. This includes things like the CSV file, the navigation software, and the augmented overlay. The CSV file is where the words in the drop down menu come from. The navigation software controls the algorithm which decides what directions the program gives the user. The Augmented overlay is the images that are placed on the screen in order to provide the user with the navigation.

|  |  |
| --- | --- |
| *Module Name* | Main |
| *Module Type* | Coordination |
| *Input Arguments* | None |
| *Output Arguments* | None |
| *Descriptions* | The main function is the entry point for the program. It starts the GUI and awaits user input. Once the input is entered it corresponds with the Background Code to implement calculations and additional work functions. |
| *Modules Invoked* | GUI, Background Code |

Table 5.1.3: Main

Table 5.1.3 shows the specifics for the Main module. The Main module controls the flow of the program. It shares information between the GUI and Background Code to make the program work. This module is designed for coordination between separate functions and to transfer information between those functions.

|  |  |
| --- | --- |
| *Module Name* | GUI |
| *Module Type* | Input and Output |
| *Input Arguments* | Design code |
| *Output Arguments* | Attractive menus and easy to use features |
| *Descriptions* | This will make the app easy to use and attractive to the user |
| *Modules Invoked* | DropDown Menu, Direction Style |

Table 5.1.4: GUI

Table 5.1.4 shows some of the specifics of the graphic user interface(GUI) module. The user interface is primarily input and output, calling the drop down menu and direction style. This module is primarily designed for attractiveness and ease of use.

|  |  |
| --- | --- |
| *Module Name* | DropDown Menu |
| *Module Type* | Input and Output |
| *Input Arguments* | User selected building/room |
| *Output Arguments* | Destination |
| *Descriptions* | Easy way for user to select their location |
| *Modules Invoked* | None |

Table 5.1.5: DropDown Menu

Table 5.1.5 shows some of the specifics of the drop down menu. The menu is primarily input and output. This module is primarily designed for ease of use, and also to allow the user to select where they are going.

|  |  |
| --- | --- |
| *Module Name* | Direction Style |
| *Module Type* | Input and Output |
| *Input Arguments* | User desired style |
| *Output Arguments* | Either the 2D TopDown styler, or the 3D Camera Overlay |
| *Descriptions* | Provides the user with two different options of how they can reach their destination |
| *Modules Invoked* | 2D TopDown, Camera Overlay |

Table 5.1.6: Direction Style

Table 5.1.6 shows some of the specifics of the direction style module. This module is primarily input and output. This module allows the user to determine whether they want a 2d or 3d display. This module invokes both of those options depending on what the user picks.

|  |  |
| --- | --- |
| *Module Name* | 2D TopDown |
| *Module Type* | Input and Output |
| *Input Arguments* | Directions |
| *Output Arguments* | 2D top-down view of building floor overlayed with directional arrow |
| *Descriptions* | Module to help user see the layout of the floor and to easily interpret directions |
| *Modules Invoked* | None |

Table 5.1.7: 2D TopDown

Table 5.1.7 shows some of the specifics of the 2D TopDown module. This module allows the user to see the map of the building from above. This module shows the directions much like Google Maps does.

|  |  |
| --- | --- |
| *Module Name* | Camera Overlay |
| *Module Type* | Input and Output |
| *Input Arguments* | Image captured from environment |
| *Output Arguments* | Display on the device’s screen |
| *Descriptions* | Captures images and displays them onto the screen of the device |
| *Modules Invoked* | None |

Table 5.1.8: Camera Overlay

Table 5.1.8 shows some of the specifics of the Camera Overlay. This module allows the user to see the image captured from the environment. This image is displayed on the device’s screen for the user.

|  |  |
| --- | --- |
| *Module Name* | Background Code |
| *Module Type* | Input and Output |
| *Input Arguments* | Images from camera, CSV file information |
| *Output Arguments* | Calculations and Directions |
| *Descriptions* | This code calculates where the user needs to go, and which direction they should take based on the captured image and input destination |
| *Modules Invoked* | CSV, Navigation, Augmented Overlay |

Table 5.1.9: Background Code

Table 5.1.9 shows some of the specifics of the Background Code. The module takes images from the camera, and the CSV information in order to develop the calculations and directions the rest of the code requires in order to develop the output of the program. This code calculates how the user gets to where they need to go. This module invokes the CSV, Navigation, and Augmented Overlay modules.

|  |  |
| --- | --- |
| *Module Name* | CSV |
| *Module Type* | Input and Output |
| *Input Arguments* | None |
| *Output Arguments* | Building/floor , Room number, and room description |
| *Descriptions* | Method of formatting the data used to find each room in a building |
| *Modules Invoked* | None |

Table 5.1.10: CSV

Table 5.1.10 shows some of the specifics of the CSV module. This module is given by the programmer/administrator and outputs the building, floor, room number, and room description. This module includes the information necessary for the drop down menu.

|  |  |
| --- | --- |
| *Module Name* | Navigation |
| *Module Type* | Input and Output |
| *Input Arguments* | Current user location |
| *Output Arguments* | Path the user should take to get to destination |
| *Descriptions* | The process of determining user position, and routing them to their destination |
| *Modules Invoked* | None |

Table 5.1.11: Navigation

Table 5.1.11 shows some of the specifics of the Navigation module. This module is given the user’s current location and takes that to develop a path that the user should take to get to their destination.

|  |  |
| --- | --- |
| *Module Name* | Augmented Overlay |
| *Module Type* | Input and Output |
| *Input Arguments* | Images from camera |
| *Output Arguments* | Overlay arrow displayed onto screen based on the users destination |
| *Descriptions* | Displays an arrow that will navigate the user to their destination |
| *Modules Invoked* | None |

Table 5.1.12: Augmented Overlay

The module shown in Table 5.1.12 is the Augmented Overlay module. The image from the camera is taken and the augmented reality (directions in the form of arrows) is overlaid on the image. This module takes the image and uses to determine where to put arrows which tell the user where they’re going.

**5.2 Detailed Design (Drawings) and Calculations**

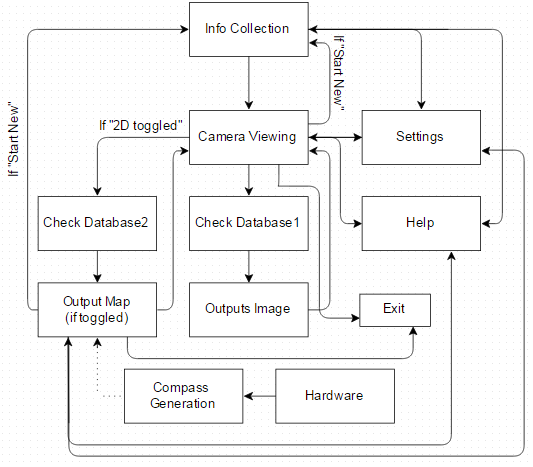
****

Figure 5.2.1 - Augmented Reality App Flow Chart

This flowchart is a graphical representation of the Augmented Reality App’s sequence of events. The App will start by collecting information from the user (described more below). The app will then use the camera to verify where the user’s current location is. From there the default setting is to check database1 for the directions and output the Augmented Arrow pointing in the direction the user must take to reach their destination. This continually loops to reassess the user’s location and output the directions on the camera screen. An optional function is to have a 2D map that the user can use to also reach their destination. Should the user toggle the 2D Map function the program will check database2 for the image corresponding to their destination and then output the image of a top down view of the building with a line connecting the user to their destination. On top of that the Team added an additional optional feature of a compass in the corner of the 2D Map to provide users with a frame of reference. When the user arrives at their destination, using either the default or 2D Map output, they will be able to exit the program or start a new iteration of the program (called “Start New”). The user will, at any time, be able to access the “Settings” or “Help” Screen through a button at the top menu. Through the “Settings” page the user will be able to control certain aspects of the program to customize their experience. The “Help” page helps the user understand the programs functionality.

|  |  |
| --- | --- |
| *Module Name* | Info Collection |
| *Module Type* | Coordination |
| *Input Arguments* | Start of Program |
| *Output Arguments* | User choices on Building Selection and Room Selection |
| *Descriptions* | The user is prompted to input the building they are in and what room they are trying to get to. |
| *Modules Invoked* | Camera Viewing |

Table 5.2.1: Info Collection

Table 5.2.1 shows the Info Collection block of the Software Flow Chart. During this function the user is asked to provide information so the App can provide directions. These inputs are Building Selection and Room Selection.

|  |  |
| --- | --- |
| *Module Name* | Camera Viewing |
| *Module Type* | Input and Output |
| *Input Arguments* | Info Collection (User choices on Building Selection/Room Selection) |
| *Output Arguments* | Current Location |
| *Descriptions* | Camera Viewing has multiple functions.   1. The user scans a room sign so the program can receive the current location. 2. It outputs the Augmented Arrow for Directions to the user’s destination. 3. It outputs the 2D Map if user selected that option (\*this option is optional for completion of this project). 4. Once the user has reached their destination it allows them to start over (Start New) to a new destination or Exit the program. |
| *Modules Invoked* | Check Database1, Check Database2, Settings, Help, Start New, Exit |

Table 5.2.2: Camera Viewing

Table 5.2.2 shows the Module for the Camera Viewing Block. This block is used for multiple functions such as scanning for current location, outputting the directions for the user, or showing the map (depending on which is toggled).

|  |  |
| --- | --- |
| *Module Name* | Check Database1 |
| *Module Type* | Database Acquirement |
| *Input Arguments* | Current Location, User choice on Building Selection/Room Selection |
| *Output Arguments* | Directions to Destination |
| *Descriptions* | Check Database1 accepts the user’s information on which building they’re in, what room they’re going to, and where they currently are and outputs the list of directions to Outputs Image. |
| *Modules Invoked* | Outputs Image |

Table 5.2.3: Check Database1

Table 5.2.3 shows the Check Database1 Module. This function acquires the information from the user and then checks, according to that information, the corresponding directions. These directions are then inserted into the Outputs Image function.

|  |  |
| --- | --- |
| *Module Name* | Outputs Image |
| *Module Type* | Input and Output |
| *Input Arguments* | Database Directions |
| *Output Arguments* | “Augmented Arrow” Directions |
| *Descriptions* | Accepts the direction inputs and outputs to Camera Viewing the Augmented Arrow to represent the actual direction. |
| *Modules Invoked* | Camera Viewing |

Table 5.2.4: Outputs Image

Table 5.2.4 shows the Module for the Outputs Image Block. This function takes the directions from Database1 and outputs them as an Augmented Arrow overlayed over the Camera Viewing function.

|  |  |
| --- | --- |
| *Module Name* | Check Database2 |
| *Module Type* | Database Acquirement |
| *Input Arguments* | Current Location, User choice on Building Selection/Room Selection |
| *Output Arguments* | Image Location |
| *Descriptions* | Outputs the Image Location to Output Map. |
| *Modules Invoked* | Output Map |

Table 5.2.5: Check Database2

Table 5.2.5 shows the Check Database2 Module. If the 2D Map is toggled then after Camera Viewing it jumps to this function. This function acquires the information from the user and then checks for the Map that will show the correct directions. The location of the correct image is output to Output Map. This Module/Function is optional and may or may not be included in the final product.

|  |  |
| --- | --- |
| *Module Name* | Hardware |
| *Module Type* | Input and Output |
| *Input Arguments* | None |
| *Output Arguments* | Compass Directions |
| *Descriptions* | The phones internal hardware. |
| *Modules Invoked* | Compass Generation |

Table 5.2.6: Hardware

Table 5.2.6 is the module for the Hardware Block. This block is used to show that the program will use the phone’s internal hardware for calculations. These calculations are output to Compass Generation. This Module/Function is optional and may or may not be included in the final product.

|  |  |
| --- | --- |
| *Module Name* | Compass Generation |
| *Module Type* | Input and Output |
| *Input Arguments* | Hardware |
| *Output Arguments* | Compass |
| *Descriptions* | Uses the phones internal hardware to generate a compass. |
| *Modules Invoked* | Linked to Output Map |

Table 5.2.7: Compass Generation

Table 5.2.7 is the Compass Generation Module. This function generates a compass using the calculations from the Hardware block. The compass is then inserted into the corner of the Map in Output Map to be used for user understanding. This Module/Function is optional and may or may not be included in the final product.

|  |  |
| --- | --- |
| *Module Name* | Output Map (if toggled) |
| *Module Type* | Input and Output |
| *Input Arguments* | Image location from Database. |
| *Output Arguments* | Building Floor Plan with Directional Line |
| *Descriptions* | If this function is toggled then it outputs the corresponding picture from the database to Camera Viewing. |
| *Modules Invoked* | Camera Viewing |

Table 5.2.8: Output Map

Table 5.2.8 shows the Module for the Output Map Block. If the 2D Map button is toggled then Output Map will be called. This function takes the location from Check Database2 and outputs a map with directions for the user to follow. This Module/Function is optional and may or may not be included in the final product.

|  |  |
| --- | --- |
| *Module Name* | Settings |
| *Module Type* | Input and Output |
| *Input Arguments* | None |
| *Output Arguments* | None |
| *Descriptions* | List of Settings that are customizable by the User. Can be accessed at any time and returns to User’s previous location. |
| *Modules Invoked* | Previous User Placement |

Table 5.2.9: Settings

Table 5.2.9 shows the Module for the Settings Block. This function provides information to the user that they can change for more customization. This Module/Function is optional and may or may not be included in the final product.

|  |  |
| --- | --- |
| *Module Name* | Help |
| *Module Type* | Input and Output |
| *Input Arguments* | None |
| *Output Arguments* | None |
| *Descriptions* | List of information regarding the app. Can be accessed at any time and returns to User’s previous location. |
| *Modules Invoked* | Previous User Placement |

Table 5.2.10: Help

Table 5.2.10 shows the Module for the Help Block. This function provides the user with helpful information (such as how the App works) and tips for using the App. A FAQ will also be provided. This Module/Function is optional and may or may not be included in the final product.

|  |  |
| --- | --- |
| *Module Name* | Exit |
| *Module Type* | Coordination |
| *Input Arguments* | None |
| *Output Arguments* | None |
| *Descriptions* | Closes the App and exits the current directional information. |
| *Modules Invoked* | None |

Table 5.2.11: Exit

Table 5.2.11 shows the Module for the Exit Block. This block exits the program and erases the current directional information. When the user opens the App again they will start over from the beginning. This Module/Function is optional and may or may not be included in the final product.

|  |  |
| --- | --- |
| *Module Name* | Start New |
| *Module Type* | Coordination |
| *Input Arguments* | None |
| *Output Arguments* | None |
| *Descriptions* | Starts App over again so User can insert a new destination. |
| *Modules Invoked* | Info Collection |

Table 5.2.12: Start New

Table 5.2.12 shows the Module for the Start New Block. This function allows the user to start over from the beginning of the App without having to close and restart it. When selected the user will jump back to the Info Collection block and will be prompted with the corrections corresponding to destination. This Module/Function is optional and may or may not be included in the final product.

**User Interface Flow Diagram**



Figure 5.2.2 - User Interface Flow Diagram

This Flow Diagram is a graphical representation of what the User will see at each stage of the Program. At the Building Selection phase the user will be prompted to enter which building they are in. To do this the user will be offered a drop down menu with the available buildings (for this project Engineering is to be completed other buildings are optional and additional work). Next the user is prompted to enter the room that they’re trying to get to. They then use the scan to scan a room sign to enter their current location and then depending on the toggle it’ll go to either the Augmented Arrow Directions or 2D Map Layout.

**5.3 Updated SOW and Project Management Plan**

Our statement of work goes through what the team is currently working on, and the current standpoint on the project’s completion.

1. Currently working on Deliverable 5
   1. Design Architecture
   2. Detailed Design
2. Performing background research to solidify our understanding of the SDK and Android programming.
3. Researching relevant technology to find out what types of features are applicable to the app, and choose which types of features we plan on implementing.
4. Learning the Javascript programming language that is associated with the Vuforia SDK.

**5.4 Detailed Project Budget**

Total Purchasing Budget: $500

Capital Equipment

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Item Description** | **Item No.** | **Company** | **URL** | **Unit price** | **Quantity** | **Subtotal** |
| Google Nexus 7 Tablet | N/A | Amazon | <http://www.amazon.com/Nexus-Google-7-Inch-Black-Tablet/dp/B00DVFLJDS/ref=sr_1_1?s=pc&ie=UTF8&qid=1415662359&sr=1-1&keywords=google+nexus+7> | 175.99 | 1 | 175.99 |
| Vuforia SDK | N/A | Qualcomm | <https://www.qualcomm.com/products/vuforia> | FREE | 1 | FREE |
| Total | 175.99 |  |  |  |  |  |
| Budget | 500 |  |  |  |  |  |
| Surplus | 324.01 |  |  |  |  |  |

Table 5.13: Capital Equipment - Bill of Materials (BOM)

Administrative Supplies: None

Reference Materials: None

Staffing

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Hours** | **Salary ($ per hour)** | **Subtotal** |
| Jeremy Fischer | 60 | 20 | 1200 |
| Dylan Shanahan | 71 | 20 | 1420 |
| Jordan Zielger | 64 | 20 | 1280 |
| Justus Karenzi | 61.5 | 20 | 1230 |
| Sudi Nsengiyumva | 60 | 20 | 1200 |
| Thomas Renck (Advisor) | 3 | 40 | 120 |
| Total |  |  | 6450 |

Table 5.14: Payroll